An insertion operator preserving infinite reduction sequences

David Chemouil

▶ **To cite this version:**

David Chemouil. An insertion operator preserving infinite reduction sequences. Mathematical Structures in Computer Science, 2008, 18 (4), pp.693-728. 10.1017/S0960129508006816 . hal-00782799

HAL Id: hal-00782799
https://hal.science/hal-00782799

Submitted on 30 Jan 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

AN INSERTION OPERATOR PRESERVING INFINITE REDUCTION SEQUENCES

DAVID CHEMOUIL

Publisher version available in *Mathematical Structures in Computer Science / Volume 18 / Special Issue 04 / August 2008, pp 693-728*, DOI: 10.1017/S0960129508006816.

ABSTRACT. A common way to show the termination of the union of two abstract reduction systems, provided both systems terminate, is to prove they enjoy a specific property (some sort of "commutation" for instance). This specific property is actually used to show that, for the union not to terminate, one out of both systems must itself be non-terminating, which leads to a contradiction. Unfortunately, the property may be impossible to prove because some of the objects that are reduced do not enjoy an adequate form.

The purpose of this article is then threefold. It first introduces an operator enabling to insert a reduction step on such an object, and therefore to change its shape, while still preserving the ability to use the property. Of course, some new properties should be verified.

Secondly, as an instance of our technique, the operator is applied to relax a well-known lemma stating the termination of the union of two termination abstract reduction systems.

Finally, this last lemma is applied in a peculiar and in a more general way to show the termination of some lambda-calculi with inductive types augmented with specific reductions dealing with: (1) copies of inductive types; and (2) with the representation of symmetric groups.

1. INTRODUCTION

1.1. **Motivation.** Given two reduction systems, one common way to prove the termination of the union of both systems goes this way:

(1) if one or both systems are themselves terminating;
(2) and some other property (*e.g.* some sort of "commutation") is verified;

then the application of an appropriate lemma concludes the expected result. This gives an instance of so-called *modular* proofs in rewriting theory.

The lemma in question is itself often proved by first supposing that the union is *not* terminating and then, using properties 1 and 2, by showing that it leads to a contradiction (typically: one can build a reduction sequence beginning by an infinite fragment of one out of both reductions, while it is already known to be terminating, by hypothesis).

The ability to build an infinite reduction sequence out of another one (also terminating) is crucial and is usually ensured by property 2. Some examples of such a property may be found in [Bachmair and Dershowitz, 1986]. One nice feature of such a proof technique is that it often relies on simple *diagram chasing*.

Sometimes, however, these proofs will not work because there are some cases where the objects (whatever they may be: terms, trees, graphs, or so) are too complex for the diagrams to be closed. On the other hand, there are situations s.t. if the object at the root of the diagram satisfied yet another condition, the diagram in question could be closed. If this condition could itself be obtained for a *reduct* of the root object, the method still works , using a slightly modified version of the original lemma with subsequent conditions.

For this, we define an operator transforming an infinite reduction sequence into another one by *inserting* a given reduction step at the beginning of the sequence. Then we give an example of how it may be used to derive the new lemma ensuring that the union of two reduction systems is terminating. Finally, we use this new lemma to redo and generalise some proofs carried out in different articles [Chemouil and Soloviev, 2003, Soloviev and Chemouil, 2004, Chemouil, 2005].

1.2. **Terminology and Notations.** Most of our terminology and notations come from [Baader and Nipkow, 1998] and [Terese, 2003] :

Definition 1.1 (Abstract Reduction System)**.** *An* Abstract Reduction System *(*ARS *for short) is a structure* $\mathcal{A} := (A, \{\rightarrow_R \mid R \in I\})$*, where* A *is a set of* objects *and* $\{\rightarrow_R \mid R \in I\}$ *is a set of binary* reduction relations *on* A *indexed by a set* I*.*

One also writes R instead of $\rightarrow_R$. Usually, the word "rewrite" is reserved for *concrete* systems, such as Term Rewrite Systems or Graph Rewrite Systems; hence we shall avoid to use it when dealing with ARS.

We will also use the following notations:

$r \rightarrow_R s$	$:=$	$(r, s) \in \rightarrow_R$	
$\leftarrow_R$	$:=$	${\rightarrow_R}^{-1}$	inversion
$\rightarrow_R \rhd \rightarrow_S$	$:=$	$\{(x, z) \mid \exists\, y \in A \cdot x \rightarrow_R y \wedge y \rightarrow_S z\}$	composition
$\overset{0}{\rightarrow}_R$	$:=$	$\{(x, x) \in A^2\}$	
$\overset{n+1}{\rightarrow}_R$	$:=$	$\overset{n}{\rightarrow}_R \rhd \rightarrow_R$	
$\overset{=}{\rightarrow}_R$	$:=$	$\rightarrow_R \cup \overset{0}{\rightarrow}_R$	reflexive closure
$\leftrightarrow_R$	$:=$	$\rightarrow_R \cup \leftarrow_R$	symmetric closure
$\overset{*}{\rightarrow}_R$	$:=$	$\bigcup_{n \geqslant 0} \overset{n}{\rightarrow}_R$	reflexive transitive closure
$\overset{+}{\rightarrow}_R$	$:=$	$\bigcup_{n \geqslant 1} \overset{n}{\rightarrow}_R$	transitive closure
$\overset{*}{\leftrightarrow}_R$	$:=$	$(\leftrightarrow_R)^*$	*convertibility*
$\rightarrow_{RS}$	$:=$	$\rightarrow_R \cup \rightarrow_S$	

Dotted lines shall also be used in diagrams in order to represent existential statements in the corresponding formulas.

If $r \rightarrow_R s$, one says that r *reduces in one step* to s and that the latter is a *one-step reduct* of the former. If $r \overset{+}{\rightarrow}_R s$, one simply says that r *reduces* to s and that the latter is a *reduct* of the former.

Given an object r, various different reductions are likely to be possible and, hence, many *reduction sequences*.

An object is a *normal form* if it is not reducible. Furthermore, an object is *terminating*[1] if *every* reduction sequence going from this object leads to a normal form.

Definition 1.2 (Termination)**.** *Given a reduction relation* R *on* A*, the set* SN *of* terminating objects *is the smallest set s.t.*

$$\forall\, r \cdot (\forall\, s \cdot r \rightarrow_R s \Rightarrow s \in \mathsf{SN}) \Rightarrow r \in \mathsf{SN} \ .$$

Thus, an object is terminating if every one-step reduct is. If an object is in normal form, the left-hand side of the implication is void and the implication is therefore verified.

As a matter of fact, a terminating object is sometimes defined by stipulating that there is no infinite reduction sequence going from it. This last notion is rather considered as "non-non-termination" by some authors but both notions can be proved equivalent using non-constructive reasoning[2].

We will write $R \vDash \Downarrow$ if every object in A is terminating for R.

Finally, in the following pages, we shall give names to some specific reduction sequences. For example, we shall write $u \in U^*$, which means u is a (possibly empty) finite sequence of pairs $(t_i, t_{i+1}) \in U$ (with $i \geqslant 0$). We shall also write $u \rhd v$ to express the fact that u and v are reduction sequences (u

[1] One also says "strongly normalising".

[2] See [Matthes, 2000], pp. 28–29, and the thread "SN" from the mailing-list "proof theory" (`list@prooftheory.org`), February 2004.

being finite) s.t. the second element of the last pair in u coincides with the first element of the first pair in v.

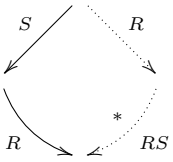
1.3. **The Adjournment Lemma.** Let us now give an example of a lemma deriving the termination of the union of two ARS. This lemma has known a few variants and various names (see [Bachmair and Dershowitz, 1986, Geser, 1990, Doornbos and von Karger, 1998, Chemouil and Soloviev, 2003]).

Definition 1.3 (Adjournment)**.** *Let R and S be two reduction relations. Then S is adjournable w.r.t. R if $S \triangleright R \subseteq R \triangleright (RS)^*$.*

$S \triangleright R \subseteq R \triangleright (RS)^*$ is the relational way to write the following statement:

$$\forall r \forall s \forall t \cdot r \to_S s \wedge s \to_R t \Rightarrow \exists u \cdot r \to_R u \wedge u \overset{*}{\to}_{RS} t \ .$$

The corresponding diagram is the following.



Lemma 1.1 (Adjournment)**.** *Let R and S be two reduction relations s.t.:*

- *$R \vDash \Downarrow$;*
- *$S \vDash \Downarrow$;*
- *S is adjournable w.r.t. R.*

Then $RS \vDash \Downarrow$.

Proof. Let R and S be two reduction relations s.t. $R \vDash \Downarrow$, $S \vDash \Downarrow$ and S is adjournable w.r.t. R. Suppose RS is not terminating. Then, there exists an infinite sequence of RS-reductions, alternating finite subsequences of R- and S-reductions, as R and S both terminate. Therefore, going from the beginning of the sequence, one can "pull up" an R-reduction by adjourning the first S-reduction followed by an R-reduction. Iterating this process yields an infinite R-reduction sequence. Contradiction. □

1.4. **Example.** Though useful, this lemma is not powerful enough to deal with some special reductions which we added to a simply-typed λ-calculus with inductive types. This is why we devised the insertion operator to be presented. But let us first give a quick glance at an example of a situation where usual results do not apply well or at all (this situation is dealt with in full detail in Section 4.3).

Our example appears in a simply-typed λ-calculus with inductive types (that is, recursive types with only strictly-positive occurrences of recursion type variables). Briefly (see Section 4.1 for full details), the β_μ-reduction rule for primitive recursion is given by:

$$(\!|\overrightarrow{t}|\!)\ (c_i\ \overrightarrow{r}) \to_{\beta_\mu} t_i\ \overrightarrow{r}\ \overrightarrow{(\!(\!|\overrightarrow{t}|\!)\ r^0)}\ \overrightarrow{((\!|\overrightarrow{t}|\!) \circ r^1)}\ ,$$

where $(\!|\overrightarrow{t}|\!)$ is the recursion operator, $(c_i\ \overrightarrow{r})$ is a constructor of an inductive type together with its parameters, r^0 is the result of recursive calls on a non-functional parameter, and r^1 is the result of recursive calls on a functional parameter.

Let us now write $[n]$ for the set $\{i \in \mathbb{N} \mid 1 \leqslant i \leqslant n\}$. The set $[n]$ is then represented by an inductive type, denoted $\overline{n} := \mu\alpha\,(c_1^{\overline{n}} : \alpha, \ldots, c_n^{\overline{n}} : \alpha)$ (there can be various representations, just by changing names of constructors, of course). Then, to every function $f : [n] \to [m]$ corresponds a term $\overline{f} : \overline{n} \to \overline{m}$ of the form $(\!|c_{f(1)}^{\overline{m}}, \ldots, c_{f(n)}^{\overline{m}}|\!)^{\overline{n},\overline{m}}$, with $\overline{m} := \mu\alpha\,(c_1^{\overline{m}} : \alpha, \ldots, c_m^{\overline{m}} : \alpha)$.

Now, suppose one wants to embed the notion of symmetric group σ_n into the reduction relation (see Section 4.3 for full details). In such a setting, it is well-known that any permutation $s \neq \mathsf{id}$ can be decomposed in a unique way (up to the ordering of factors) in a product of cycles with mutually disjoint carriers. Composition of disjoint cycles is commutative but it remains possible to set up an ordering $<$,

lexicographical for instance, on cycles in order to obtain a canonical decomposition. Hence, every map $f : [n] \to [n]$ ($n \geqslant 2$) is equal to a product of m mutually disjoint cycles ($m \geqslant 2$) $f_1, \ldots, f_m$:

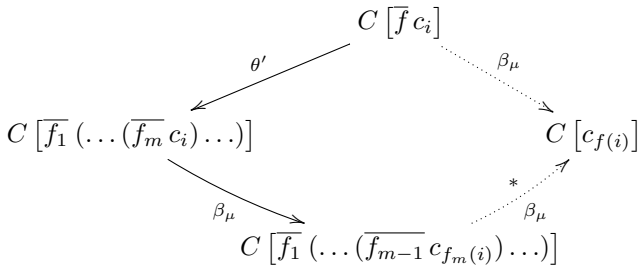
$$f = f_1 \circ \cdots \circ f_m \quad \text{(with } f_1 < \ldots < f_m\text{)}.$$

This obviously leads to defining a so-called θ'-reduction by

$$\overline{f}\, r \to_{\theta'} \overline{f_1}\, (\ldots (\overline{f_m}\, r) \ldots)$$

for every permutation $f \in \sigma_n$, for all $n \geqslant 2$, where f is decomposed into $m \geqslant 2$ mutually disjoint cycles $\{f_1, \ldots, f_m\} \subseteq \sigma_n$.

Then, to show termination of $\beta\eta\theta'$-reduction (which includes $\beta_\mu\theta'$ reduction as well as other usual reduction rules, such as $\beta_\to$-reduction and $\eta_\to$-expansion), there is a difficulty: if we try to adjourn θ' w.r.t. $\beta\eta$, we get the following diagram for β_μ-reduction:



which does not enable us to state much.

Now, let us call $\beta_{\overline{n}}$-reduction the fragment of β_μ applying to terms $\overline{f} : \overline{n} \to \overline{n}$; we then have:

$$\overline{f}\, c_i^{\overline{n}} \to_{\beta_{\overline{n}}} c_{f(i)}^{\overline{n}} \quad \text{for all } \overline{f} : \overline{n} \to \overline{n}.$$

One can then notice that adjournment would be usable provided terms were initially in $\beta_{\overline{n}}$-normal form, first. There lies the idea of the operator devised in this paper which precisely enables to change the shape of objects, while preserving infinite reduction sequences dealt with in proofs of lemmas such as the Adjournment Lemma.

1.5. **Outline of the Paper.** In Section 2, we define the operator inserting a reduction step at the beginning of a reduction sequence, yielding a new one which is proved to be infinite if the first one was. We also give some simple sufficient conditions when the operator verifies a certain property.

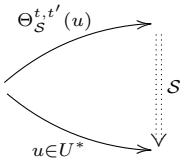
Then, in Section 3, we define a specialised form of the Adjournment Lemma enabling to deal with some complex situations.

Finally, in Section 4, we show how this new lemma was used on two situations regarding the addition of some extensional rewrite rules in a simply-typed λ-calculus with inductive types. A first exposition of these situations can be found in [Chemouil, 2005] and [Soloviev and Chemouil, 2004] without as much abstraction and as many details as in the present article.

2. An Insertion Operator

2.1. **Intuition.** Let us first give some intuition. Let U and T be ARS. We are going to define an *insertion* operator, inserting a given T-reduction (t, t') at the beginning of a possibly infinite sequence of U-reductions (provided a few properties are verified, of course), thus yielding a new sequence. Using a particular relation $\mathcal{S}$ on objects from both sequences, the operator will ensure that the sequence shall be infinite if the initial sequence was.

Intuitively, the idea underlying our technique, for a finite sequence, is the following: given a sequence $u \in U^*$, we build a new sequence $\Theta_{\mathcal{S}}^{t,t'}(u) \in U^*$ in such a way that, to every reduction step in u, there corresponds *some* reduction(s) in $\Theta_{\mathcal{S}}^{t,t'}(u)$ after which it is possible to "find again" the original sequence u through $\mathcal{S}$. Because of this idea of *return* to the initial sequence, we point this relation from the new sequence back to the initial (this is expressed using an *oriented* arrow in the accompanying diagram).



Indeed, inserting a T-reduction at the beginning of u implies that many descendants of the object reduced by T are likely to be in a shape, in $\Theta_{\mathcal{S}}^{t,t'}(u)$, different than the shape of the descendants of the same object not reduced initially, in u. The relation $\mathcal{S}$ enables to draw a correspondence between the descendants in both sequences.

2.2. **The Lemma.**

Definition 2.1 (Prosimulation)**.** *Let U be an* ARS *and $\mathcal{S}$ be a binary relation on the domain of U. Then $\mathcal{S}$* prosimulates *U if $\mathcal{S} \rhd \rightarrow_U \subseteq \overset{*}{\rightarrow}_U \rhd \mathcal{S}$.*

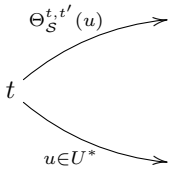
Notice that this definition is correct as the domain of $\overset{*}{\rightarrow}_U$ is the same as the one of U.

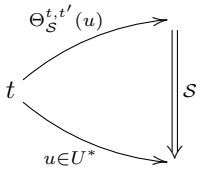
Choosing the letter $\mathcal{S}$ for this relation is not fortuitous: indeed, this is to recall the notion of *simulation* from concurrency theory. In this discipline, a simulation is a relation between labelled transition systems and provides a behavioural abstraction of such a system. In our context, $\mathcal{S}$ is a simulation from U to U^*. Note that some similar questions related to simulation were also studied independently in [Lengrand, 2005].

Definition 2.2 (Insertion Operator)**.** *Let U, T be* ARS*, $\mathcal{S}$ a relation, u a finite sequence of U-reductions beginning with an object t and (t, t') a T-reduction, s.t.:*

- *$\mathcal{S}$ prosimulates U;*
- *and $(t', t) \in \mathcal{S}$.*

We define an insertion operator $\Theta_{\mathcal{S}}^{t,t'}$ *which, from the sequence u, yields a new sequence $\Theta_{\mathcal{S}}^{t,t'}(u)$ beginning with the T-reduction (t, t'), and ensuring that:*

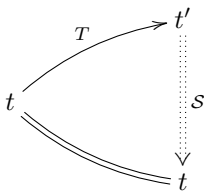
(I1) the diagram 

can be closed by $\mathcal{S}$, that is: 

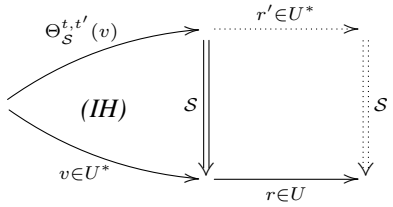
(I2) and $\Theta_{\mathcal{S}}^{t,t'}(u \rhd r) = \Theta_{\mathcal{S}}^{t,t'}(u) \rhd r'$ where $r' \in U^*$.

The sequence $\Theta_{\mathcal{S}}^{t,t'}(u)$ is defined by recursion on u:

- *If u is empty, we insert the T-reduction: $\Theta_{\mathcal{S}}^{t,t'}(u) := (t, t') \in T$. As $(t', t) \in \mathcal{S}$, we have:*



- *Otherwise, $u = v \rhd r$ with $v \in U^*$ and $r \in U$. Then, we have $\Theta_{\mathcal{S}}^{t,t'}(u) := \Theta_{\mathcal{S}}^{t,t'}(v) \rhd r'$ with:*



Remark 2.1. *As for the last case, note that we only use this definition for deterministic or finite and bounded cases, hence the case when there exist arbitrarily many r' such that $\Theta_{\mathcal{S}}^{t,t'}(u) := \Theta_{\mathcal{S}}^{t,t'}(v) \rhd r'$ will not occur.*

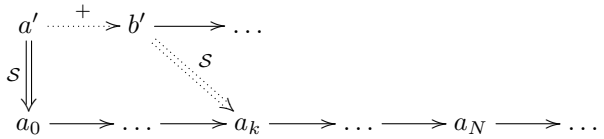
Notice that properties (I1) and (I2) are verified in both cases. In particular, the second property enables us to extend the operator to infinite reduction sequences: indeed, appending a new reduction step to an initial finite sequence keeps unchanged the reduction sequence corresponding to the initial fragment.

As will be made clearer later, the aim of this operator is just to constructively assess the existence of an infinite sequence given another one.

Definition 2.3 (Echo)**.** *Let $\to$ be a reduction relation and $\mathcal{S}$ a relation on objects. Then $\mathcal{S}$* echoes *$\to$ if*

$$\forall\, a_0 \cdot \forall\, a' \; \mathcal{S}\; a_0 \cdot \exists\, N \in \mathbb{N}^* \cdot \forall\, a_0 \to \cdots \to a_N \cdot \exists\, k \in \{1, \ldots, N\} \cdot \exists\, b' \; \mathcal{S}\; a_k \cdot a' \xrightarrow{+} b' \ .$$

In other words, $\mathcal{S}$ echoes $\to$ if, going from an object a_0 and any object a' s.t. $a'\ \mathcal{S}\ a_0$, we can find a bound $N > 0$ s.t., for every finite fragment of length N of *any* sequence of reductions (possibly infinite) beginning by a_0, there is an object a_k in this fragment (with $k \geqslant 1$) with an object b' s.t. $b'\ \mathcal{S}\ a_k$ which, itself, derives from a' in *at least one* step. This is informally expressed in the following diagram:



Intuitively, one can see that if the initial sequence is infinite, the resulting one will also be infinite, as echoing ensures that, for some finite fragments of sufficient length in the initial sequence, the resulting sequence will also contain reduction steps.

Lemma 2.1 (Insertion)**.** *Let U, T be* ARS *and $\mathcal{S}$ a relation s.t. $\mathcal{S}$ prosimulates and echoes U. Then, for any infinite sequence u of U-reductions, beginning by t, and every T-reduction (t, t') s.t. $(t', t) \in \mathcal{S}$, the sequence $\Theta_{\mathcal{S}}^{t,t'}(u)$ is also infinite.*

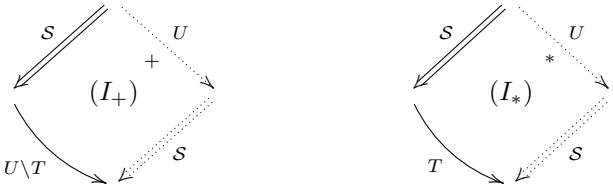
Proof. Let u be an infinite sequence of U-reductions beginning by an object t_0. As $\mathcal{S}$ prosimulates U, we can build $\Theta_{\mathcal{S}}^{t_0,t'}(u)$. This sequence cannot end as, because $\mathcal{S}$ echoes U, there necessarily exists a bound to the length of initial fragments of u for which there is at least a reduction step in $\Theta_{\mathcal{S}}^{t_0,t'}(u)$. This process of stepping along u and finding corresponding steps in $\Theta_{\mathcal{S}}^{t_0,t'}(u)$ can be iterated infinitely as u is not terminating. □

2.3. **Some Sufficient Conditions.** We now discuss the case when U enjoys specific properties, then when $\mathcal{S}$ is not only an arbitrary relation but a reduction relation itself.

2.3.1. *Insertability and Echoing.* Suppose we have $T \subsetneq U$. The insertion operator is initially defined for finite sequences, but so as to be extendible to non-terminating ones. As a consequence, as we want the sequence $\Theta_{\mathcal{S}}^{t,t'}(u)$ to be infinite provided the sequence u is, two cases are possible:

- In the case where a reduction in u comes from $U \setminus T$, it should be echoed by at least one U-reduction.
- On the other hand, if it is a T-reduction, there could even be no corresponding U-reduction because inserting a T-reduction at the very beginning of $\Theta_{\mathcal{S}}^{t,t'}(u)$ implies that, perhaps, the T-reduction which stood in u is not needed anymore at the same time in $\Theta_{\mathcal{S}}^{t,t'}(u)$.

Furthermore, it is important to be ever able to "come back" to u through $\mathcal{S}$. It is then necessary to ensure that the following diagrams can be closed.



Definition 2.4 (Insertability)**.** *Let U, T be* ARS *and $\mathcal{S}$ a binary relation on the domain of U. Then T* can be inserted in U w.r.t. $\mathcal{S}$ *if:*

- $T \subsetneq U$ *;*
- (I_+) $\quad \mathcal{S} \rhd\!\!\rightarrow_U \setminus \rightarrow_T \subseteq \overset{+}{\rightarrow}_U \rhd \mathcal{S}$*;*
- (I_*) $\quad \mathcal{S} \rhd\!\!\rightarrow_T \subseteq \overset{*}{\rightarrow}_U \rhd \mathcal{S}$.

Lemma 2.2. *Let U, T be* ARS *and $\mathcal{S}$ a binary relation on the domain of U s.t.:*

- *T is finitely branching ;*
- *T can be inserted in U w.r.t. $\mathcal{S}$;*
- *$T \vDash \Downarrow$.*

Then $\mathcal{S}$ prosimulates and echoes U.

Proof. First, T can be inserted in U w.r.t. $\mathcal{S}$, therefore $\mathcal{S}$ obviously prosimulates U. Furthermore, by König's Lemma, the fact that T is finitely branching and terminating implies it is always possible to find, for every initial term, the bound necessary to echoing. □

Corollary 2.1. *Let U, T be* ARS *and $\mathcal{S}$ a binary relation on the domain of U s.t.:*

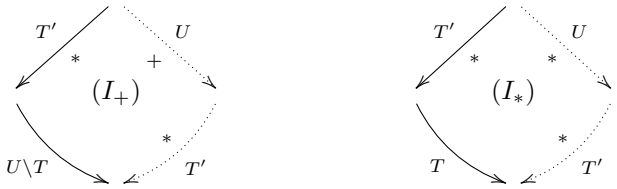
- *T can be inserted in U w.r.t. $\mathcal{S}$;*
- *$T \vDash \Downarrow$.*

Then, for any infinite sequence u of U-reductions, beginning by t, and every T-reduction (t, t') s.t. $(t', t) \in \mathcal{S}$, the sequence $\Theta_{\mathcal{S}}^{t,t'}(u)$ is also infinite.

2.3.2. *Case where the Prosimulation is a Reduction.* The situation can be even more particular. Indeed, $\mathcal{S}$ may be a reduction sequence itself, from terms appearing in $\Theta_{\mathcal{S}}^{t,t'}(u)$ to those in u. For instance, we shall use the operator to insert an $\eta_{\rightarrow}$-expansion, in Section 4.2. The relation $\mathcal{S}$ could be defined as the inverse of an $\eta_{\rightarrow}$-expansion. Unfortunately, it is not always obvious to define the inverse of such a reduction as it is context sensitive. But we know that $\eta_{\rightarrow}^{-1} \subseteq \overline{\eta}_{\rightarrow}$ (where $\overline{\eta}_{\rightarrow}$ is $\eta_{\rightarrow}$*-contraction*) and this is therefore the relation we shall use for $\mathcal{S}$. (Nevertheless, considering a reduction relation for $\mathcal{S}$ is not an obligation as we shall see in Section 4.3.)

Generally, as we insert a T-reduction at the beginning of the sequence, it will be necessary to be able to come back to the initial sequence by "anti-reducing" the descendants of the subterm which was T-reduced initially. As these descendants may enjoy several occurrences, we shall therefore consider a relation T'

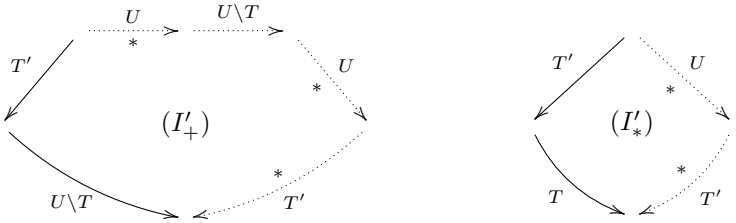
such that $T^{-1} \subseteq T'$ (in other words: $(t', t) \in T'$) and we shall take T'^* for $\mathcal{S}$. In this case, the Lemma 2.2 imposes to solve diagrams of the following shape:



As such, the lemma is not easy to use because of the universal quantification on arbitrary-length sequences of T'-reductions. The following sufficient conditions are general enough and much simpler to prove.

Lemma 2.3.
$T' \rhd (U \setminus T) \subseteq U^* \rhd (U \setminus T) \rhd U^* \rhd T'^* \quad \wedge \quad T' \rhd T \subseteq U^* \rhd T'^* \quad \Rightarrow \quad (I_+) \wedge (I_*).$

Proof. We respectively write (I'_+) and (I'_*) for the two members from the left-hand side of the implication.



Notice first that

$$(I'_+) \quad \Rightarrow \quad T' \rhd (U \setminus T) \subseteq U^* \rhd T'^* \ ,$$

hence

$$(I'_*) \quad \wedge \quad (I'_+) \quad \Rightarrow \quad T' \rhd U \subseteq U^* \rhd T'^* \ .$$

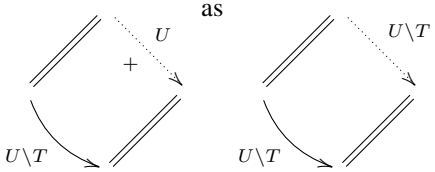
Writing $(*)$ for $T'^* \rhd U^* \subseteq U^* \rhd T'^*$, we obviously have (by induction)

$$T' \rhd U \subseteq U^* \rhd T'^* \quad \Rightarrow \quad (*) \ .$$

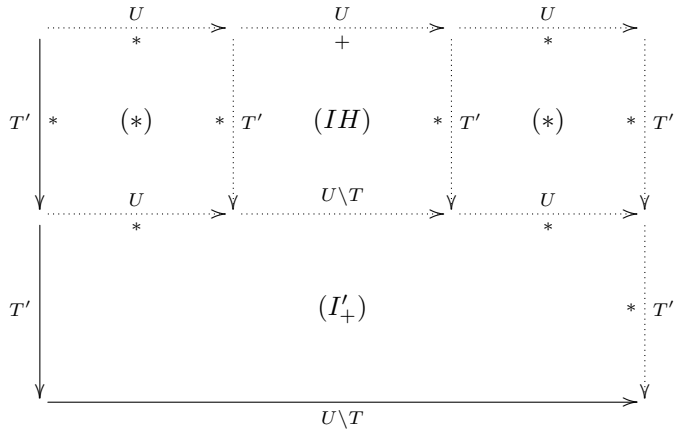
then $(I'_*) \quad \wedge \quad (*) \quad \Rightarrow \quad (I_*)$.

We now show that $(I'_+) \wedge (*) \Rightarrow (I_+)$ by induction on the length of T'^*:

- if the length is null, we obviously have as



- otherwise the sequence is of the form $T'^* \rhd T'$ and we have



□

Corollary 2.2. *Let U, T and T' be* ARS *s.t.:*

- $T \subsetneq U$;
- $T \vDash \Downarrow$;
- $T^{-1} \subseteq T'$;
- $T' \rhd (U \setminus T) \subseteq U^* \rhd (U \setminus T) \rhd U^* \rhd T'^*$;
- $T' \rhd T \subseteq U^*$.

Then, for any infinite sequence u of U-reductions, beginning by t, and every T-reduction (t, t'), the sequence $\Theta_S^{t,t'}(u)$ is also infinite.

3. An Application of Insertion in Rewriting: Pre-Adjusted Adjournment

It is rather usual in rewriting theory and λ-calculus to fall upon theorems stating the termination of a reduction relation by supposing first the existence of an infinite sequence of reductions and by showing then that it leads to a contradiction. The proof of the Adjournment Lemma (Lemma 1.3) gives a perfect example of such a situation.

In this kind of cases, if the premises for the definition of the insertion operator are met, it is possible to insert a finite number of reductions ahead of the initial infinite sequence, still preserving the non-termination and thus the possibility to find a contradiction. We shall see in section 4.2 that adjournment alone does not suffice to prove the termination of the union of two particular relations because there is a case where a term must be $\eta_\rightarrow$-expanded beforehand in a certain way. Our insertion operator enables us precisely to insert these necessary $\eta_\rightarrow$-expansions. In this particular case, $\mathcal{S}$ is the $\eta_\rightarrow$*-contraction* which enables to return to the initial sequence. We shall also use the operator in its full generality (*i.e.* without $\mathcal{S}$ being a reduction relation) in section 4.3.

Definition 3.1 (Conditional Adjournment)**.** *Let R and S be* ARS *and P a predicate on objects. Then S is* adjournable w.r.t. R under condition P *if*

$$\forall\, a \forall\, b \forall\, c \cdot P(a) \wedge a \rightarrow_S b \wedge b \rightarrow_R c \Rightarrow \exists\, d \cdot a \rightarrow_R d \stackrel{*}{\rightarrow}_{RS} c \ .$$

The necessity of conditional adjournment became clear to us during our study of non-standard reductions in calculi with inductive types. We think that in case of non-standard reductions, for example, representing mathematical content of the theories formulated in Logical Frameworks it will be rather a rule that ordinary methods based on permutation or ordinary adjournment of conversions turn out to be insufficient.

Definition 3.2 (Realisation)**.** *Let T be an* ARS*, P a predicate on objects and a and object. Then T* realises *P for a if $\exists\, b \cdot a \xrightarrow{*}_T b \wedge P(b)$. We shall also say that T* realises *P if T realises P for any object a.*

Lemma 3.1 (Pre-Adjusted Adjournment)**.** *Let R, S, and T be* ARS*, $\mathcal{S}$ a relation and P a predicate on objects s.t.:*

- *S is adjournable w.r.t. R under condition P ;*
- *$T \subseteq R$;*
- *$R \vDash \Downarrow$;*
- *$S \vDash \Downarrow$;*
- *T realises*[3] *P ;*
- *$\mathcal{S}$ prosimulates RS ;*
- *$\mathcal{S}$ echoes RS.*

Then $RS \vDash \Downarrow$.

Proof. By Lemma 2.1, there exists $\Theta_{\mathcal{S}}^{t,t'}$ that preserves infinite sequences of R-reductions beginning by t, for any T-reduction (t, t') s.t. $(t', t) \in \mathcal{S}$.

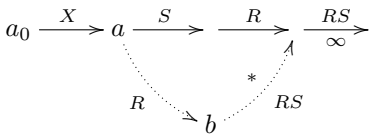
Suppose now RS is not terminating, *i.e.* there is (at least) one infinite sequence of RS-reductions. Take any sequence beginning by an object a_0. As R and S are terminating, we can deduce that this sequence is made up from an infinite alternation of finite fragments of R- and S-reductions.

Now, we apply the following process: we follow the sequence from the beginning until reaching the first occurrence of an S-reduction immediately followed by an R-reduction, for an object a. In other words, we have initially a finite fragment X of reductions, where $X = R^*S^*$ (that is $X = R^*$ or $X = R^*S^+$), leading to a. The sequence is therefore of the following form

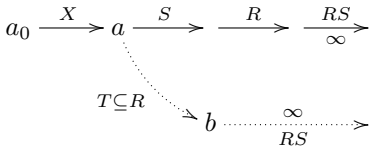
$$a_0 \xrightarrow{X} a \xrightarrow{S} \xrightarrow{R} \xrightarrow[\infty]{RS} \quad .$$

There are now two possibilities regarding a:

- Either $P(a)$: it is then possible to adjourn the S-reduction following a, thus obtaining:



- Or $\neg P(a)$: it is then possible to insert a T-reduction from a, keeping a infinite sequence of RS-reductions:



Now, if $X = R^*$, the new infinite sequence begins with a fragment of R-reductions longer than in the initial sequence (as $T \subseteq R$). It is therefore possible to iterate this process from b, and to keep lengthening the initial fragment of R-reductions. If $X = R^*S^+$, it is necessary to iterate this process once more from a_0 because the new sequence is of the form

$$a_0 \xrightarrow[*]{R} \xrightarrow[*]{S} c \xrightarrow{S} \xrightarrow{R} b \xrightarrow[\infty]{RS}$$

and a new conditional adjournment (possibly foregone by insertions to realise P for c) is therefore necessary to extend the initial fragment of R-reductions.

[3] In fact, as the following proof shows, it would even be enough to state that: for any object b, if $\neg P(b)$ then b is not T-normal.

In both cases, the initial fragment of S-reductions gets smaller and we build a fragment of R-reductions which is longer and longer, which is an infinite process as RS does not terminate. But $R \vDash \Downarrow$. □

4. Applications in Lambda-Calculus

We now present two applications of our Pre-Adjusted Adjournment Lemma. They were first presented in [Soloviev and Chemouil, 2004, Chemouil, 2005] but very briefly, due to lack of space. These studies motivated the creation of the insertion operator. We will now enter into more details.

4.1. **The Setting.** Let us first recall the setting of our work, which is a simply-typed λ-calculus featuring arrow, product and unit types, all together with computational and extensional rewrite rules, as well as inductive types (*i.e.* some special case of more general recursive types) with computational rewrite rules corresponding to structural recursion.

Definition 4.1 (Types)**.** *Define, by mutual induction:*

- *the set* T *of types*

$$\frac{\alpha \in \mathsf{TV}}{\alpha \in \mathsf{T}}\ (V) \qquad \frac{}{1 \in \mathsf{T}}\ (1) \qquad \frac{\rho \in \mathsf{T} \qquad \sigma \in \mathsf{T}}{(\rho \times \sigma) \in \mathsf{T}}\ (\times)$$

$$\frac{\rho \in \mathsf{T} \qquad \sigma \in \mathsf{T}}{(\rho \to \sigma) \in \mathsf{T}}\ (\to) \qquad \frac{\overrightarrow{\kappa} \neq \varnothing \qquad \overrightarrow{c} \subseteq \mathsf{C} \qquad \alpha \in \mathsf{TV} \qquad \overrightarrow{\kappa} \subseteq \mathsf{Sch}(\alpha)}{\mu\alpha\,(\overrightarrow{c} : \overrightarrow{\kappa}) \in \mathsf{T}}\ (\mu)$$

- *the set* $\mathsf{Sch}(\alpha)$ *of (constructor) schemas on a type variable* α

$$\frac{\overrightarrow{\rho} \subseteq \mathsf{T} \qquad \overrightarrow{\rho} \subseteq \mathsf{SPos}(\alpha)}{\overrightarrow{\rho} \to \alpha \in \mathsf{Sch}(\alpha)}\ (\textsc{Sch})$$

- *the set* $\mathsf{SPos}(\alpha)$ *of strictly positive operators over a type variable* α

$$\frac{\rho \in \mathsf{T} \qquad \alpha \notin \mathsf{FV}(\rho)}{\rho \in \mathsf{SPos}(\alpha)}\ (\textsc{SPosPar}) \qquad \frac{\overrightarrow{\rho} \subseteq \mathsf{T} \qquad \alpha \notin \mathsf{FV}(\overrightarrow{\rho})}{\overrightarrow{\rho} \to \alpha \in \mathsf{SPos}(\alpha)}\ (\textsc{SPosRec})$$

- *and the set of free type variables (defined the usual way).*

One can remark that a schema κ on α is written $\rho_1 \to \ldots \to \rho_m \to \alpha$ ($m \geqslant 0$). If $m = 0$, the schema is called *empty*. If it is non-empty, every ρ_j is either a *strictly-positive* operator where α is not free; or a strictly-positive of the form $\sigma_1 \to \ldots \to \sigma_p \to \alpha$ (with $1 \leqslant j \leqslant m$). In the first case, we speak of a *parametric operator* and, in the second case, of a *0-recursive operator* if $\overrightarrow{\sigma} = \varnothing$ or of a *1-recursive operator* otherwise (by analogy with Gödel's finite type recursive functionals).

Example 4.1. *Essentially, the strictly-positive approach for inductive types enables to define infinitely branching trees with finite depth. A typical example is given by the representation* $\mathbf{O}$ *of Brouwer's ordinals (where* $\mathbf{N}$ *is the inductive type of natural numbers):*

$$\mathbf{O} := \mu\alpha\,(0 : \alpha, S : \alpha \to \alpha, L : (\mathbf{N} \to \alpha) \to \alpha)\ .$$

Here, α *is an empty schema,* $\alpha \to \alpha$ *is 0-recursive and* $(\mathbf{N} \to \alpha) \to \alpha$ *is 1-recursive.*

Definition 4.2 (Preterms)**.** *The set of preterms is generated by the following grammar:*

$$t ::= x \mid \star \mid \langle t, t\rangle^{\rho\times\sigma} \mid \mathsf{p}_1{}^{\rho\times\sigma} t \mid \mathsf{p}_2{}^{\rho\times\sigma} t \mid \lambda x^{\rho} t \mid (t\,t) \mid (c\,\overrightarrow{t}) \mid (\!|\,\overrightarrow{t}\,|\!)^{\mu\alpha\,(\overrightarrow{c}:\overrightarrow{\kappa}),\rho}\ ,$$

with $c \in \mathsf{C}$ *and* $x \in \mathsf{V}$.

Preterms of the form $(\!|\overrightarrow{t}|\!)^{\mu\alpha\,(\overrightarrow{c}:\overrightarrow{\kappa}),\rho}$ stand for *recursors* in the sense of Gödel's System T. They contain as many *recursion terms* as their domain contains constructors. On the other hand, preterms of the form $c\,\overrightarrow{t}$ inhabit an inductive type. Finally, p_1 and p_2 are projections (we will sometimes write p_i to speak about any projection). We now introduce typing the usual way.

Definition 4.3 (Recursion Type)**.** *Given a constructor $c \in \hat{\mu}$ with schema $\overrightarrow{\rho} \to \alpha$ and any type σ, the recursion type $\delta_c^{\hat{\mu},\sigma}$ of c is defined by:*

$$\delta_c^{\hat{\mu},\sigma} := \overrightarrow{\rho}[\alpha := \hat{\mu}] \to \overrightarrow{\|\rho\|}[\alpha := \sigma] \to \sigma \ ,$$

where

$$\|\rho\| := \begin{cases} \varnothing & \text{if } \rho \text{ is a parametric operator,} \\ \rho & \text{if } \rho \text{ is a recursive operator.} \end{cases}$$

Example 4.2. *Recursion types for Brouwer's ordinals, to any type σ, are:*

$$\begin{array}{lcl} \delta_0^{\mathbf{O},\sigma} & = & \sigma \\ \delta_S^{\mathbf{O},\sigma} & = & \mathbf{O} \to \sigma \to \sigma \\ \delta_L^{\mathbf{O},\sigma} & = & (\mathbf{N} \to \mathbf{O}) \to (\mathbf{N} \to \sigma) \to \sigma \ . \end{array}$$

Definition 4.4 (Typing)**.** *The typing relation $\Gamma \vdash t : \rho$ is defined by induction on preterms:*

$$\frac{(x,\rho) \in \Gamma}{\Gamma \vdash x : \rho}\ (V) \qquad \frac{}{\Gamma \vdash \star : 1}\ (1\text{-I})$$

$$\frac{\Gamma \vdash r : \rho \qquad \Gamma \vdash s : \sigma}{\Gamma \vdash \langle r,s\rangle^{\rho\times\sigma} : \rho \times \sigma}\ (\times\text{-I}) \qquad \frac{\Gamma \vdash r : \rho \times \sigma}{\Gamma \vdash (\mathsf{p}_1{}^{\rho\times\sigma} r) : \rho}\ (\times\text{-E}_1) \qquad \frac{\Gamma \vdash r : \rho \times \sigma}{\Gamma \vdash (\mathsf{p}_2{}^{\rho\times\sigma} r) : \sigma}\ (\times\text{-E}_2)$$

$$\frac{\Gamma, x : \rho \vdash r : \sigma}{\Gamma \vdash (\lambda x^{\rho} r) : \rho \to \sigma}\ (\to\text{-I}) \qquad \frac{\Gamma \vdash r : \rho \to \sigma \qquad \Gamma \vdash s : \rho}{\Gamma \vdash (rs) : \sigma}\ (\to\text{-E})$$

$$\frac{(c, \overrightarrow{\rho} \to \alpha) \in \hat{\mu} \qquad \Gamma \vdash \overrightarrow{r} : \overrightarrow{\rho}[\alpha := \hat{\mu}]}{\Gamma \vdash (c\overrightarrow{r}) : \hat{\mu}}\ (\mu\text{-I}) \qquad \frac{\Gamma \vdash \overrightarrow{t} : \overrightarrow{\delta_c^{\hat{\mu},\sigma}}}{\Gamma \vdash (\!|\overrightarrow{t}|\!)^{\hat{\mu},\sigma} : \hat{\mu} \to \sigma}\ (\mu\text{-E})$$

Definition 4.5 (Terms)**.** *A* term *is a well-typed preterm.*

We now introduce the reduction relation operating on the basic calculus. The computational aspect is expressed by β-reduction while the extensional one is expressed by η-reduction. First of all, given two terms $g : \sigma \to \tau$ and $f : \overrightarrow{\rho} \to \sigma$, we write $g \circ f := \lambda\overrightarrow{x}^{\overrightarrow{\rho}} \cdot g\,(f\,\overrightarrow{x})$.

Definition 4.6 (β-reduction)**.** *We define* β-reduction *the following way:*

$$\begin{array}{lll} (\lambda x r)s & \to_{\beta_\to} & r[x := s] \\ \mathsf{p}_1\langle r,s\rangle & \to_{\beta_{\times_1}} & r \\ \mathsf{p}_2\langle r,s\rangle & \to_{\beta_{\times_2}} & s \\ (\!|\overrightarrow{t}|\!)\ (c_i\,\overrightarrow{r}) & \to_{\beta_\mu} & t_i\,\overrightarrow{r}\,\overrightarrow{\Delta_r} \end{array}$$

where

$$\Delta_r := \begin{cases} (\!|\overrightarrow{t}|\!)\ r & \text{if the operator corresponding to } r \text{ is 0-recursive,} \\ (\!|\overrightarrow{t}|\!) \circ r & \text{if the operator corresponding to } r \text{ is 1-recursive,} \\ \varnothing & \text{otherwise.} \end{cases}$$

Example 4.3. *Here are the rules for β_μ-reduction on Brouwer's ordinals:*

$$\begin{array}{lll}(\!|t_0, t_S, t_L|\!)\ 0 & \to_{\beta_\mu} & t_0\\ (\!|t_0, t_S, t_L|\!)\ (S\ p) & \to_{\beta_\mu} & t_S\ p\ ((\!|t_0, t_S, t_L|\!)\ p)\\ (\!|t_0, t_S, t_L|\!)\ (L\ k) & \to_{\beta_\mu} & t_L\ k\ ((\!|t_0, t_S, t_L|\!) \circ k)\ .\end{array}$$

To simplify matters, we shall order operators so that every canonical object $c\ \overrightarrow{r}$ inhabiting an inductive type is of the form $c\ \overrightarrow{r^p}\ \overrightarrow{r^0}\ \overrightarrow{r^1}$, where:

- operators corresponding to $\overrightarrow{r^p}$ are parametric ;
- operators corresponding to $\overrightarrow{r^0}$ are 0-recursive ;
- operators corresponding to $\overrightarrow{r^1}$ are 1-recursive.

Hence β_μ-reduction can be written:

$$(\!|\overrightarrow{t}|\!)\ (c_i\ \overrightarrow{r}) \to_{\beta_\mu} t_i\ \overrightarrow{r}\ \overrightarrow{((\!|\overrightarrow{t}|\!)\ r^0)}\ \overrightarrow{((\!|\overrightarrow{t}|\!) \circ r^1)}\ .$$

Is is now common to define η-reduction as an *expansive* rule. Though it is then context dependent, its metatheory is usually nicer than for η-contraction which does not behave well when combined with a rule for the unit type (see [Di Cosmo, 1995, Di Cosmo, 1996a, Di Cosmo and Kesner, 1993, Di Cosmo and Kesner, 1996a, Di Cosmo and Kesner, 1996b, Di Cosmo, 1996b]).

Definition 4.7 (η-reduction)**.** *Define η-reduction by:*

$$r \to_{\eta_\to} \lambda x^\rho \cdot rx \quad \text{if} \begin{cases} r : \rho \to \sigma, \\ x \notin \mathsf{FV}(r), \\ r \text{ is not an abstraction nor in applicative position.}\end{cases}$$

$$r \to_{\eta_\times} \langle \mathsf{p}_1 r, \mathsf{p}_2 r\rangle \quad \text{if} \begin{cases} r \text{ is of product type,} \\ r \text{ is not a pair nor projected.}\end{cases}$$

$$r \to_{\eta_1} \star \quad \text{if} \begin{cases} r : 1, \\ r \neq \star.\end{cases}$$

Theorem 4.1. $\beta\eta \vDash \Downarrow \wedge\ \beta\eta \vDash \diamond$.

Proof. $\beta \vDash \Downarrow$ and $\beta \vDash \diamond$ are proven using respectively the Tait-Girard method [Girard et al., 1988] and the Tait-Martin-Löf method [Barendregt, 1984] and Newman's Lemma. Then η-reductions are added using standard abstract techniques such as Akama-Di Cosmo's Lemma [Di Cosmo, 1996b]. □

(In [Soloviev and Chemouil, 2004, Chemouil, 2005], we also have a so-called ν-reduction which implements extensionality on inductive representations of the product and unit types: adding this reduction also yields a convergent system. However, this reduction is not important regarding the topic of the present article, so we will not consider it here.)

4.2. **Copies.** We now rework the notion of copy of an inductive type introduced in [Chemouil, 2005] using the specialised Pre-Adjusted Adjournment Lemma.

4.2.1. *Isomorphisms of Types.* We first define the notion of *isomorphism of types:* we need a typed λ-calculus together with an equivalence relation $\sim$ on terms, an associative composition operator $\circ_{\rho,\sigma,\tau} : (\sigma \to \tau) \to (\rho \to \sigma)$ (simply written $\circ$) and a term $\mathsf{id}_\rho : \rho \to \rho$ neutral for $\circ$ (for all types ρ, σ, τ). Here, $\circ$ and id_ρ can be defined the obvious way, but not necessarily.

Definition 4.8 (Isomorphism of Types)**.** *Two types ρ and σ are* isomorphic*, written $\rho \cong \sigma$, if there exist two terms $f : \rho \to \sigma$ and $g : \sigma \to \rho$ s.t. $f \circ g \sim \mathsf{id}_\sigma$ and $g \circ f \sim \mathsf{id}_\rho$.*

It is important to notice that an isomorphism between types might be *provable* but not *computable*. This is the reason why it is necessary to devise a rewriting relation implementing $\sim$ and prove its termination and confluence.

4.2.2. *Taxonomy.*

Definition 4.9 (Copy)**.** *Let there be two types π and π' and two inductive types φ and φ' s.t. if π appears in φ, it is only as a parameter or as the full domain of the functional argument of a 1-recursive operator. Let there also be two terms $f : \pi \to \pi'$ and $f' : \pi' \to \pi$ (to take into account co- and contravariant arguments).*

Then, φ' is a copy *of φ induced by f and f' if the first type only differs from the second one by constructor names and by the fact that zero or several occurrences of π in φ are replaced by π' in φ'.*

If we have a *computable* isomorphism $f : \pi \cong \pi' : f'$, $\mu\alpha\,(\overrightarrow{c} : \overrightarrow{\kappa})$ and $\mu\alpha\,(\overrightarrow{c'} : \overrightarrow{\kappa'})$ are called *faithful copy* of one another. If π does not appear in $\mu\alpha\,(\overrightarrow{c} : \overrightarrow{\kappa})$ or if no occurrence of it is replaced (in other words: φ and φ' only differ by constructor names), we shall speak of *carbon copy*. We then have

$$\text{carbon copy} \subseteq \text{faithful copy} \subseteq \text{copy}.$$

In this article, we shall be at best interested in faithful copies.

Remark 4.1. *Notice that we will not consider possible constructor reorderings, here. On the contrary, we shall consider that every c_i in $\mu\alpha\,(\overrightarrow{c} : \overrightarrow{\kappa})$ corresponds precisely to c'_i in $\mu\alpha\,(\overrightarrow{c'} : \overrightarrow{\kappa'})$.*

Obviously, faithful copies form *provable* isomorphisms. We shall make them *computably* isomorphic by adding an adequate reduction (χ).

4.2.3. *Maximal Abstracted Form and Choice Representation.* This section is here to introduce 2 technical definitions which shall be useful later.

Definition 4.10 (Maximal Abstracted Form)**.** *Given a term r, we call* maximal abstracted form *of r the term written $\lceil r \rceil$ s.t. $\lceil r \rceil$ begins by as many λ-abstractions as the arity of r.*

Remark that $r \xrightarrow{*}_{\eta_\to} \lceil r \rceil$. However, the maximal abstracted form of a term may differ from its $\eta_\to$-normal form because the strict subterms of r may not be in normal form in the first case.

We aim at defining terms fc and fc' realising the mapping from a type to another one, as long as the latter is a faithful copy of the former, induced by terms f and f'. To do so, we present here a notation enabling to precise which occurrence of a parameter if modified. Implicitly, we imagine a user has "selected" or "chosen" previously these occurrences.

Definition 4.11 (Choice)**.** *We shall write $\underline{f}\, r$ for $f r$ if $r : \pi$ corresponds to an occurrence to be replaced, and for r otherwise.*

Similarly, we shall write $g \circ \underline{f}$ for $g \circ f$ if g has domain π and must be replaced, and for g otherwise.

Remark 4.2. *This notation will only be used in particular contexts, to obtain subterms $\underline{f'}(\underline{f}\, r)$ or $g \circ \underline{f} \circ \underline{f'}$ all the time, for terms f and f' s.t. $f : \pi \cong \pi' : f'$.*

Notice that $\underline{f'}(\underline{f}\,r) = r$ *and* $g \circ \underline{f} \circ \underline{f'} = g$ *if no replacement is carried out. Else, as* f *and* f' *are computable isomorphisms by the relation* $\to_R$*, we have* $\underline{f'}(\underline{f}\,r) = f'(fr) \overset{*}{\to}_R r$ *and* $g \circ \underline{f} \circ \underline{f'} = g \circ f \circ f' \overset{*}{\to}_R \lceil g \rceil$.

4.2.4. *Realising Faithful Copies.* We shall write $\mathsf{fc} : \mu\alpha\,(\overrightarrow{c} : \overrightarrow{\kappa}) \to \mu\alpha\,(\overrightarrow{c'} : \overrightarrow{\kappa'})$ and $\mathsf{fc}' : \mu\alpha\,(\overrightarrow{c'} : \overrightarrow{\kappa'}) \to \mu\alpha\,(\overrightarrow{c} : \overrightarrow{\kappa})$ for terms generated from two faithful copies and terms $f : \pi \cong \pi' : f$. Informally, making a difference between parametric, 0-recursive and 1-recursive arguments, we shall have:

$$\mathsf{fc}\,(c_i\ \overrightarrow{r^p}\ \overrightarrow{r^0}\ \overrightarrow{r^1}) = c'_i\ \overrightarrow{(\underline{f}\ r^p)}\ \overrightarrow{(\mathsf{fc}\ r^0)}\ \overrightarrow{(\mathsf{fc} \circ r^1 \circ \underline{f'})}\ .$$

We can see that choices may only appear in parameters or 1-recursive arguments as π is necessarily different from the recursion variable in inductive types.

Example 4.4. *Suppose we have a type* $\mathbf{P}$ *isomorphic to* $\mathbf{N}$ *through* $f : \mathbf{N} \cong \mathbf{P} : f'$ *and the following two inductive types:*

$$\begin{aligned}\varphi &:= \mu\alpha\,(c_1 : \alpha, c_2 : ((\mathbf{N} \to \mathbf{N}) \to \alpha) \to \alpha, c_3 : \mathbf{N} \to \alpha \to \alpha, c_4 : (\mathbf{N} \to \alpha) \to \alpha)\\ \varphi' &:= \mu\alpha\,(c'_1 : \alpha, c'_2 : ((\mathbf{N} \to \mathbf{N}) \to \alpha) \to \alpha, c'_3 : \mathbf{P} \to \alpha \to \alpha, c'_4 : (\mathbf{P} \to \alpha) \to \alpha)\ .\end{aligned}$$

We decide to replace some occurrences of the parameter $\mathbf{N}$ *by* $\mathbf{P}$*. The definition of the isomorphism* $\mathsf{fc} : \varphi \to \varphi'$ *is then:*

$$\begin{aligned}\mathsf{fc}\ c_1 &:= c'_1\\ \mathsf{fc}\ (c_2\ k) &:= c'_2\ k\\ \mathsf{fc}\ (c_3\ h\ t) &:= c'_3\ (f\ h)\ (\mathsf{fc}\ t)\\ \mathsf{fc}\ (c_4\ k) &:= c'_4\ (\mathsf{fc} \circ k \circ f')\ .\end{aligned}$$

As we write in Definition 4.9, the replaced parameter for a 1-recursive operator must be the full domain of the functional argument. For this reason, we could only make a replacement for c_2 *if we considered* $\mathbf{N} \to \mathbf{N}$ *as a parameter, and not* $\mathbf{N}$ *as it the case here. If one wishes to modify also this part, it is necessary to make a copy in two steps, that is by defining a sequence of copies* $\varphi \rightsquigarrow \varphi' \rightsquigarrow \varphi''$*, with*

$$\varphi'' := \mu\alpha\,(c_1 : \alpha, c_2 : ((\mathbf{P} \to \mathbf{P}) \to \alpha) \to \alpha, c_3 : \mathbf{P} \to \alpha \to \alpha, c_4 : (\mathbf{P} \to \alpha) \to \alpha)\ .$$

Definition 4.12. *Let* $\varphi := \mu\alpha\,(\overrightarrow{c} : \overrightarrow{\kappa})$ *and* $\varphi' := \mu\alpha\,(\overrightarrow{c'} : \overrightarrow{\kappa'})$ *be faithful copies induced by* $f : \pi \cong \pi' : f'$*. Benefiting from the ordering convention stated on p. 13, every constructor* c_i *has type* $\kappa_i[\alpha := \varphi] = \overrightarrow{\tau} \to \overrightarrow{\varphi} \to \overrightarrow{(\overrightarrow{\sigma} \to \varphi)} \to \varphi$*, where* $\overrightarrow{\tau}$ *corresponds to parameters,* $\overrightarrow{\varphi}$ *to 0-recursive operators and* $\overrightarrow{(\overrightarrow{\sigma} \to \varphi)}$ *to 1-recursive operators.*

We define the term $\mathsf{fc} : \varphi \to \varphi'$ *(and similarly* fc'*) by*

$$\mathsf{fc} := (\!|\overrightarrow{a}|\!)^{\varphi,\varphi'}$$

where every a_i *is the* $\to_{\beta\eta}$*-normal form of*

$$\lambda\overrightarrow{p}^{\overrightarrow{\tau}}\lambda\overrightarrow{z}^{\overrightarrow{\varphi}}\lambda\overrightarrow{u}^{\overrightarrow{\overrightarrow{\sigma}\to\varphi}}\lambda\overrightarrow{r}^{\overrightarrow{\varphi}}\lambda\overrightarrow{s}^{\overrightarrow{\overrightarrow{\sigma}\to\varphi'}} \cdot c'_i\ \overrightarrow{(\underline{f}\,p)}\ \overrightarrow{r}\ \overrightarrow{(s \circ \underline{f'})}\ .$$

Remark that fc and fc' are *closed* terms, because f and f' are (being isomorphisms).

Lemma 4.1. *φ and φ' are provably isomorphic.*

Proof. For every canonical object $c_i\ \overrightarrow{r^p}\ \overrightarrow{r^0}\ \overrightarrow{r^1}$, we have indeed $\mathsf{fc}'\ (\mathsf{fc}\ x) \overset{*}{\leftrightarrow}_{\beta\eta} x$:

$$\begin{aligned}
\mathsf{fc}'\ (\mathsf{fc}\ (c_i\ \overrightarrow{r^p}\ \overrightarrow{r^0}\ \overrightarrow{r^1})) &\to_{\beta_\mu} \mathsf{fc}'\ (a_i\ \overrightarrow{r^p}\ \overrightarrow{r^0}\ \overrightarrow{r^1}\ \overrightarrow{(\mathsf{fc}\ r^0)})\ \overrightarrow{(\mathsf{fc} \circ r^1)}) \\
&\overset{*}{\to}_{\beta_\to} \mathsf{fc}'\ (c_i'\ \overrightarrow{(\underline{f}\ r^p)}\ \overrightarrow{(\mathsf{fc}\ r^0)}\ \overrightarrow{(\mathsf{fc} \circ r^1 \mathbin{\underline{\circ} f'})}) \\
&\to_{\beta_\mu} a_i'\ \overrightarrow{(\underline{f}\ r^p)}\ \overrightarrow{(\mathsf{fc}\ r^0)}\ \overrightarrow{(\mathsf{fc} \circ r^1 \mathbin{\underline{\circ} f'})}\ \overrightarrow{(\mathsf{fc}'\ (\mathsf{fc}\ r^0))}\ \overrightarrow{(\mathsf{fc}' \circ \mathsf{fc} \circ r^1 \mathbin{\underline{\circ} f'})} \\
&\overset{*}{\to}_{\beta_\to} c_i\ \overrightarrow{(\underline{f'}\ (\underline{f}\ r^p))}\ \overrightarrow{(\mathsf{fc}'\ (\mathsf{fc}\ r^0))}\ \overrightarrow{(\mathsf{fc}' \circ \mathsf{fc} \circ r^1 \mathbin{\underline{\circ} f'} \mathbin{\underline{\circ} f})}
\end{aligned}$$

We now use the fact that $f'\ (f x) \overset{*}{\leftrightarrow}_{\beta\eta} x$:

$$\overset{*}{\leftrightarrow}_{\beta\eta} c_i\ \overrightarrow{r^p}\ \overrightarrow{(\mathsf{fc}'\ (\mathsf{fc}\ r^0))}\ \overrightarrow{(\mathsf{fc}' \circ \mathsf{fc} \circ r^1)}$$

By (IH), we notice the particular shape of the last arguments (cf. subsection 4.2.5):

$$\begin{aligned}
&\overset{*}{\leftrightarrow}_{\beta\eta} c_i\ \overrightarrow{r^p}\ \overrightarrow{r^0}\ \overrightarrow{\lceil r^1 \rceil} \\
&\leftarrow_{\eta_\to} c_i\ \overrightarrow{r^p}\ \overrightarrow{r^0}\ \overrightarrow{r^1}\ .
\end{aligned}$$

□

As this isomorphism is provable but not computable, we propose to embed it into the calculus in the following way.

Definition 4.13 (χ-reduction)**.** *Define* χ-rewriting *by:*

$$\begin{array}{ll} (\chi_1) & \mathsf{fc}'\ (\mathsf{fc}\ r) \to_\chi r \\ (\chi_2) & \mathsf{fc}\ (\mathsf{fc}'\ r) \to_\chi r\ , \end{array}$$

and χ-reduction *as its contextual closure.*

Lemma 4.2 (Substitutivity and Compatibility)**.**

- $r \to_\chi r' \Rightarrow r[x := s] \to_\chi r'[x := s]$ *;*
- $s \to_\chi s' \Rightarrow r[x := s] \overset{*}{\to}_\chi r[x := s']$.

Proof. Easy inductions. □

4.2.5. *Convergence.* We must now show the convergence of $\beta\eta\chi$-reduction. Unfortunately, it does not seem possible to use a simulation or Akama-Di Cosmo's Lemma. Still, we wish to use abstract techniques, like adjournment.

The Adjournment Lemma seems adequate to show the termination, but suppose we have a term

$$(\!|\ \overrightarrow{t}\ |\!)\ (\mathsf{fc}'\ (\mathsf{fc}\ (c_i\ \overrightarrow{r^p}\ \overrightarrow{r^0}\ \overrightarrow{r^1})))$$

and following reductions:

$$(\!|\ \overrightarrow{t}\ |\!)\ (\mathsf{fc}'\ (\mathsf{fc}\ (c_i\ \overrightarrow{r^p}\ \overrightarrow{r^0}\ \overrightarrow{r^1}))) \to_\chi (\!|\ \overrightarrow{t}\ |\!)\ (c_i\ \overrightarrow{r^p}\ \overrightarrow{r^0}\ \overrightarrow{r^1}) \to_{\beta_\mu} t_i\ \overrightarrow{r^p}\ \overrightarrow{r^0}\ \overrightarrow{r^1}\ \overrightarrow{((\!|\ \overrightarrow{t}\ |\!) r^0)}\ \overrightarrow{((\!|\ \overrightarrow{t}\ |\!) \circ r^1)}\ .$$

Adjourning the χ-reduction w.r.t. the $\beta\eta$-reduction means looking for a term s s.t.

$$(\!|\ \overrightarrow{t}\ |\!)\ (\mathsf{fc}'\ (\mathsf{fc}\ (c_i\ \overrightarrow{r^p}\ \overrightarrow{r^0}\ \overrightarrow{r^1}))) \to_{\beta_\mu} s \overset{*}{\to}_{\beta\eta\chi} t_i\ \overrightarrow{r^p}\ \overrightarrow{r^0}\ \overrightarrow{r^1}\ \overrightarrow{((\!|\ \overrightarrow{t}\ |\!) r^0)}\ \overrightarrow{((\!|\ \overrightarrow{t}\ |\!) \circ r^1)}\ ,$$

because the only other possible reduction in that case is β_μ. But the sequence we end up with is rather of the following form:

$$\begin{aligned}(\!|\overrightarrow{t}|\!)\ (\mathsf{fc}'\ (\mathsf{fc}\ (c_i\ \overrightarrow{r^p}\ \overrightarrow{r^0}\ \overrightarrow{r^1}))) &\to_{\beta_\mu} \overset{*}{\to}_{\beta\eta\nu} (\!|\overrightarrow{t}|\!)\ (c_i\ \overrightarrow{r^p}\ \overrightarrow{r^0}\ \overrightarrow{\lceil r^1\rceil})\\ &\overset{*}{\to}_{\beta\eta\nu} t_i\ \overrightarrow{r^p}\ \overrightarrow{r^0}\ \overrightarrow{\lceil r^1\rceil}\ \overrightarrow{((\!|\overrightarrow{t}|\!)\ r^0)}\ \overrightarrow{((\!|\overrightarrow{t}|\!)\circ\lceil r^1\rceil)}\\ &\overset{*}{\to}_{\beta_\to} t_i\ \overrightarrow{r^p}\ \overrightarrow{r^0}\ \overrightarrow{\lceil r^1\rceil}\ \overrightarrow{((\!|\overrightarrow{t}|\!)\ r^0)}\ \overrightarrow{((\!|\overrightarrow{t}|\!)\circ r^1)}\ .\end{aligned}$$

As a consequence, if copies do not contain 1-recursive operator, the proof by adjournment is straightforward. Otherwise, one can see in the previous example that adjournment would have been possible *provided* the 1-recursive arguments had been in maximal abstracted form. This is what we are going to do using our Pre-Adjusted Adjournment Lemma.

Lemma 4.3. $\chi \vDash \Downarrow \wedge\ \chi \vDash \diamond$.

Proof. Obvious. □

We shall now use the insertion operator in the case where the relation $\mathcal{S}$ is a reduction relation containing the inverse of $\eta_\to$:

Definition 4.14 ($\overline{\eta}_\to$-reduction)**.** *We define* $\overline{\eta}_\to$-rewriting *(or* $\eta_\to$-contraction*) by*

$$\lambda x^\rho \cdot rx \to_{\overline{\eta}_\to} r \quad \textit{if } x \notin \mathsf{FV}(r),$$

and $\overline{\eta}_\to$-reduction *as its contextual closure.*

Lemma 4.4. *$\eta_\to$ can be inserted $\beta\eta\chi$ w.r.t. $\overline{\eta}_\to$.*

Proof. First of all, we obviously have $\eta_\to \subsetneq \beta\eta\chi$. By Lemma 2.3, we now have to show

$$\to_{\overline{\eta}_\to} \rhd \to_{\beta\eta_{\times,1}\chi} \subseteq \overset{*}{\to}_{\beta\eta\chi} \rhd \to_{\beta\eta_{\times,1}\chi} \rhd \overset{*}{\to}_{\beta\eta\chi} \rhd \overset{*}{\to}_{\overline{\eta}_\to}$$

and

$$\to_{\overline{\eta}_\to} \rhd \to_{\eta_\to} \subseteq \overset{*}{\to}_{\beta\eta\chi} \rhd \overset{*}{\to}_{\overline{\eta}_\to}\ .$$

(see Appendix A for relevant diagram chasing). □

Lemma 4.5. $\beta\eta\chi \vDash \Downarrow$.

Proof. We define a condition P on terms s.t. 1-recursive arguments be in maximal abstracted form. One has:

- $\eta_\to \subsetneq \beta\eta$;
- $\beta\eta \vDash \Downarrow$ by Theorem 4.1;
- $\chi \vDash \Downarrow$ by Lemma 4.3;
- $\eta_\to$ realises P as 1-recursive arguments are not in applicative position;
- $\eta_\to$ can be inserted in $\beta\eta\chi$ w.r.t. $\overline{\eta}_\to$ by Lemma 4.4 and $\eta_\to \vDash \Downarrow$. Hence, by Lemma 2.2, $\overline{\eta}_\to$ prosimulates and echoes $\beta\eta\chi$.

Moreover, as $\eta_\to^{-1} \subseteq \overline{\eta}_\to$, we know that for any pair $(t, t') \in \eta_\to$, we have indeed $(t', t) \in \overline{\eta}_\to$. Using the Pre-Adjusted Adjournment Lemma, we are left with proving that χ can be adjourned w.r.t. $\beta\eta$ under condition P (see Appendix B for relevant diagram chasing). □

Lemma 4.6. $\beta\eta\chi \vDash \diamond$.

Proof. By Lemma 4.5 and Newman's Lemma, it is enough to show that $\beta\eta\chi \vDash \mathsf{LC}$. As $\beta\eta \vDash \diamond$ and $\chi \vDash \diamond$, there is only left to show that $\leftarrow_{\beta\eta} \rhd \to_\chi \subseteq \overset{*}{\to}_{\beta\eta\chi} \rhd \overset{*}{\leftarrow}_{\beta\eta\chi}$. We do not detail fully this easy proof as confluence is not the focus of this article.

□

4.3. **The Symmetric Group.** Our purpose in [Soloviev and Chemouil, 2004] was to add some other extensional rewrite rules on inductive types corresponding to some extensional relations between inductive types (see [Chemouil, 2005] for a quite detailed explanation of the problem).

As a matter of fact, one purpose was to embed the notion of symmetric group into the reduction relation. This study began as part of the project 'Type Theory and Formal Calculus', sponsored by Liapunov Institute. The idea, presented in [Flegontov and Soloviev, 2002], consisted in formalising groups acting on some differential equations into a proof assistant. Equations were represented as vectors of parameters (seen as elements of an inductive type) and the group itself as symmetries acting on coefficients.

Let us first write $[n]$ for the set $\{i \in \mathbb{N} \mid 1 \leqslant i \leqslant n\}$. The set $[n]$ is then represented by an inductive type, denoted $\overline{n} := \mu\alpha\,(c_1^{\overline{n}} : \alpha, \ldots, c_n^{\overline{n}} : \alpha)$ (there can be various representations, just by changing names of constructors, of course). Then, to every function $f : [n] \to [m]$ corresponds a term $\overline{f} : \overline{n} \to \overline{m}$ of the form $(\!|c_{f(1)}^{\overline{m}}, \ldots, c_{f(n)}^{\overline{m}}|\!)^{\overline{n},\overline{m}}$, with $\overline{m} := \mu\alpha\,(c_1^{\overline{m}} : \alpha, \ldots, c_m^{\overline{m}} : \alpha)$.

Consider now the case of a map $f : [n] \to [n]$ and the associated term $\overline{f} : \overline{n} \to \overline{n}$. It is then possible to study the representation of the symmetric group σ_n in type theory. Let us first recall some basics of Group Theory.

Definition 4.15 (Group)**.** *A* group $(G, *, \cdot^{-1}, 1)$ *is a set G together with a binary map $*$, a unary map $\cdot^{-1}$ and an element $1 \in G$ s.t.:*

- $*$ *is associative;*
- 1 *is neutral for $*$;*
- *every element x has a symmetric x^{-1} (*i.e. *s.t. $\forall\, x \in G \cdot x * x^{-1} = x^{-1} * x = 1$).*

Definition 4.16 (Permutation)**.** *A* permutation *is a bijection from a finite set to itself.*

The *carrier of a permutation* on a set is made of those elements which aren't invariant under the permutation.

Lemma 4.7 (Symmetric Group)**.** *The set of permutations on $[n]$, together with composition as the product, is a group, called the* symmetric group of order n *and denoted σ_n.*

Definition 4.17 (Cycle)**.** *A* cycle *on a set E is a permutation s s.t. there is $\{a_1, \ldots, a_n\} \subseteq E$ s.t.*

$$\begin{array}{lcll} s(a_i) & = & a_{i+1} & \text{for } i < n; \\ s(a_n) & = & a_1 & \\ s(b) & = & b & \text{for } b \notin \{a_1, \ldots, a_n\}. \end{array}$$

Theorem 4.2. *Every permutation $s \neq \mathsf{id}$ can be decomposed in a unique way (up to the ordering of factors) in a product of cycles with mutually disjoint carriers.*

Composition of disjoint cycles is commutative but it remains possible to set up an ordering $<$, lexicographical for instance, on cycles in order to obtain a canonical decomposition. Hence, every map $f : [n] \to [n]$ ($n \geqslant 2$) is equal to a product of m mutually disjoint cycles ($m \geqslant 2$) $f_1, \ldots, f_m$:

$$f = f_1 \circ \cdots \circ f_m \quad (\text{with } f_1 < \ldots < f_m).$$

This obviously leads to the following:

Definition 4.18 (θ'-reduction)**.** *We define* θ'-reduction *by*

$$\overline{f}\, r \to_{\theta'} \overline{f_1}\, (\ldots (\overline{f_m}\, r) \ldots)$$

for every permutation $f \in \sigma_n$, for all $n \geqslant 2$, where f is decomposed into $m \geqslant 2$ mutually disjoint cycles $\{f_1, \ldots, f_m\} \subseteq \sigma_n$.

Lemma 4.8 (Substitutivity and compatibility)**.**

- $r \to_{\theta'} r' \Rightarrow r[x := s] \to_{\theta'} r'[x := s]$;
- $s \to_{\theta'} s' \Rightarrow r[x := s] \xrightarrow{*}_{\theta'} r[x := s']$.

Proof. Easy inductions. □

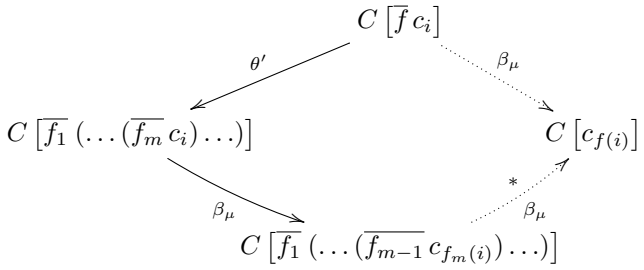
Lemma 4.9. $\theta' \vDash \Downarrow \wedge\ \theta' \vDash \diamond$.

Proof. Obvious. □

Let us call $\beta_{\overline{n}}$-reduction the fragment of β_μ applying to terms $\overline{f} : \overline{n} \to \overline{n}$; we then have:

$$\overline{f}\, c_i^{\overline{n}} \to_{\beta_{\overline{n}}} c_{f(i)}^{\overline{n}} \quad \text{for all } \overline{f} : \overline{n} \to \overline{n}.$$

Then, to show termination of $\beta\eta\theta'$-reduction, there is a difficulty: if we try to adjourn θ' w.r.t. $\beta\eta$, we get the following diagram for β_μ-reduction:



This suggests to use pre-adjusted adjournment using as a condition the fact that terms must be in $\beta_{\overline{n}}$-normal form. The lemma must be used in full generality (*i.e.* not w.r.t. a reduction). Notice that it would be hard to consider a deterministic reduction relation $\beta'_{\overline{n}}$ containing the inverse of $\beta_{\overline{n}}$. Indeed, the inverse image of a constructor c_i by the relation $\beta_{\overline{n}}$ is generally of cardinality strictly greater than 1. Intuitively, $\beta_{\overline{n}}$ is not injective.

Nevertheless, after a first observation of the situation, we can see that the inverse image by $\beta_{\overline{n}}$ is finite and that $\beta_{\overline{n}}$-redices are closed terms: therefore, it remains possible to build systematically $\mathcal{S}$ by *choosing* a term $\overline{g}$ s.t. $\overline{g}\, c_i \to_{\beta_{\overline{n}}} c_{f(i)}$. This shall only collapse, be duplicated or be reduced (even after the decomposition of $\overline{g}$ itself) to $c_{f(i)}$.

It is this remark which motivated the definition of insertion as not necessarily depending on a reduction relation.

Definition 4.19. *Let f be a permutation in σ_n. We now define a binary relation $\mathcal{S}_f$ on terms the following way. Given two terms a and a', tag with different numbers every subterm of a' of the form $c_{f(i)}$, for all i, and denote by $L_{a'}$ the list of these labelled constructors.*

Then $a'\ \mathcal{S}_f\ a$ if there exists a substitution ζ, mapping a labelled constructor to a term, s.t.:

- *$a'\zeta = a$;*
- *and, for all $c^\ell_{f(i)} \in L_{a'}$, one has $c^\ell_{f(i)}\zeta \xrightarrow{*}_{\beta_{\overline{n}}} c_{f(i)}$ (where ℓ is a label).*

It is now important to notice that it is not enough to show that $\beta_{\overline{n}}$-reduction can be inserted in $\beta\eta\theta'$ w.r.t. $\mathcal{S}_f$. Indeed, a $\beta_{\overline{n}}$-redex happens to be *also* a θ'-redex. Write $\theta' \sqcap \beta_{\overline{n}}$ for the set of θ'-reductions whose left-hand sides are $\beta_{\overline{n}}$-redices, that is of the form $\overline{f}\ c_i$ (obviously $\theta' \sqcap \beta_{\overline{n}} \subseteq \theta'$).

Lemma 4.10. $\beta_{\overline{n}} \cup (\theta' \sqcap \beta_{\overline{n}}) \vDash \Downarrow$.

Proof. It is enough to show that $\beta_{\overline{n}}\theta' \vDash \Downarrow$, which can be deduced from $\beta_{\overline{n}} \vDash \Downarrow$, $\theta' \vDash \Downarrow$ and the fact that $\to_{\beta_{\overline{n}}} \subseteq \to_{\theta'} \rhd \xrightarrow{+}_{\beta_{\overline{n}}}$. □

Lemma 4.11. *(i) $\mathcal{S}_f$ prosimulates $\beta\eta\theta'$;*
(ii) $\mathcal{S}_f$ echoes $\beta\eta\theta'$.

Proof. (i) Obvious.

(ii) We must show that

$$\forall\, a_0 \cdot \forall\, a'\ \mathcal{S}_f\ a_0 \cdot \exists\, N \in \mathbb{N}^* \cdot \forall\, a_0 \to_{\beta\eta\theta'} \cdots \to_{\beta\eta\theta'} a_N \cdot \exists\, k \in \{1, \ldots, N\} \cdot \exists\, b'\ \mathcal{S}_f\ a_k \cdot a' \xrightarrow{+}_{\beta\eta\theta'} b' \ .$$

Let a_0 and a' be s.t. $a'\ \mathcal{S}_f\ a_0$. In other words, there is a substitution ζ mapping a term to a labelled constructor s.t. $a'\zeta = a_0$ and, for all $c^\ell_{f(i)} \in L_{a'}$, we have $c^\ell_{f(i)}\zeta \xrightarrow{*}_{\beta_{\overline{n}}} c_{f(i)}$.

Now, consider all possible reduction sequences beginning by a_0 and taken from $\beta_{\overline{n}} \cup (\theta' \sqcap \beta_{\overline{n}})$. As this latter relation terminates (by Lemma 4.10), there exists a bound — let us write it N_0 — to the length of these sequences. Define $N := N_0 + 1$.

Let there be an arbitrary sequence of length N of the shape $a_0 \to_{\beta\eta\theta'} \cdots \to_{\beta\eta\theta'} a_N$. We must find a number $k \in \{1, \ldots, N\}$ s.t. there is a b' s.t. $b'\ \mathcal{S}_f\ a_1$ and $a' \xrightarrow{+}_{\beta\eta\theta'} b'$. There are two possibilities:

- Either we have $a_0 \to_{\beta\eta\theta' \setminus \beta_{\overline{n}}} a_1$: in this situation, it is enough to put $k = 1$ and to apply the same reduction on a' thus obtaining b' s.t. $b'\ \mathcal{S}_f\ a_1$. One should notice here that the value of N is not important when such a reduction is applied to a_0. This comes from the fact that subterms of a_0 in the image of ζ are $\beta_{\overline{n}}$-redices and therefore, here, can only collapse or be duplicated (in the general case, they may also enjoy a $\beta_{\overline{n}}$-reduction). The echoing is therefore obviously determined.
- Or, we have $a_0 \to_{\beta_{\overline{n}} \cup (\theta' \sqcap \beta_{\overline{n}})} a_1$. In that case, we can glance through the sequence $a_0 \to_{\beta_{\overline{n}} \cup (\theta' \sqcap \beta_{\overline{n}})} \cdots \to_{\beta\eta\theta'} a_N$ until reaching the first occurrence of a reduction different from $\beta_{\overline{n}}$ and $\theta' \sqcap \beta_{\overline{n}}$. It necessarily exists as $N > N_0$: its ordering number is the sought k and it is enough to echo it the same way than in the previous case.

□

Lemma 4.12. $\beta\eta\theta' \vDash \Downarrow$.

Proof. We define a condition P on terms s.t. these are in $\beta_{\overline{n}}$-normal form. We have:

- $\beta_{\overline{n}}\theta' \subsetneq \beta\eta$;
- $\beta\eta \vDash \Downarrow$ by Theorem 4.1 ;
- $\theta' \vDash \Downarrow$ by Lemma 4.9 ;
- $\beta_{\overline{n}}\theta'$ obviously realises P (in fact, $\beta_{\overline{n}}$ is enough) ;
- $\mathcal{S}_f$ prosimulates and echoes $\beta\eta\theta'$ by Lemma 4.11.

By the Pre-Adjusted Adjournment Lemma (Lemma 3.1), we just need to show that θ' is adjournable w.r.t. $\beta\eta$ under condition P (see Appendix C for relevant diagram chasing). □

Lemma 4.13. $\beta\eta\theta' \vDash \diamond$.

Proof. As $\beta\eta\theta' \vDash \Downarrow$ and $\beta\eta \vDash \diamond$ and $\theta' \vDash \diamond$ by Lemmas 4.12 and 4.9 and Theorem 4.1, it is enough to show that $\beta\eta$ and θ' locally commute. We do not detail this easy proof here as confluence is not the focus of this article. □

Appendix A. Proof of Lemma 4.4

Lemma 4.4. $\eta_\to$ can be inserted $\beta\eta\chi$ w.r.t. $\overline{\eta}_\to$.

Proof. First of all, we obviously have $\eta_\to \subsetneq \beta\eta\chi$. By Lemma 2.3, we now have to show

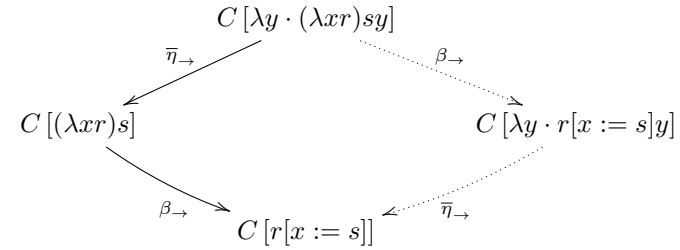
$$\to_{\overline{\eta}_\to} \rhd \to_{\beta\eta_{\times,1}\chi} \subseteq \xrightarrow{*}_{\beta\eta\chi} \rhd \to_{\beta\eta_{\times,1}\chi} \rhd \xrightarrow{*}_{\beta\eta\chi} \rhd \xrightarrow{*}_{\overline{\eta}_\to}$$

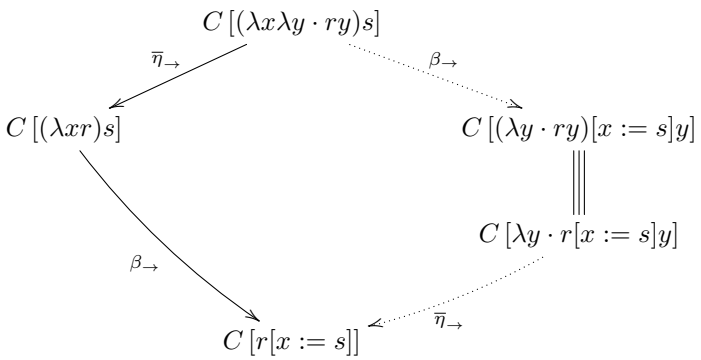
and

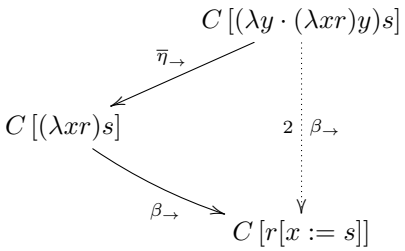
$$\to_{\overline{\eta}_\to} \rhd \to_{\eta_\to} \subseteq \xrightarrow{*}_{\beta\eta\chi} \rhd \xrightarrow{*}_{\overline{\eta}_\to} \ .$$

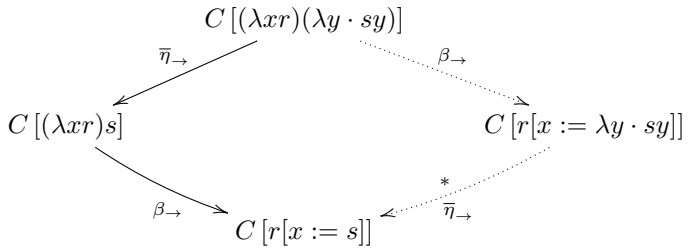
We do so by considering the possible overlaps for a context $C\,[]$, without detailing too much this demonstration which is quite simple:

- $\beta_{\to}$:

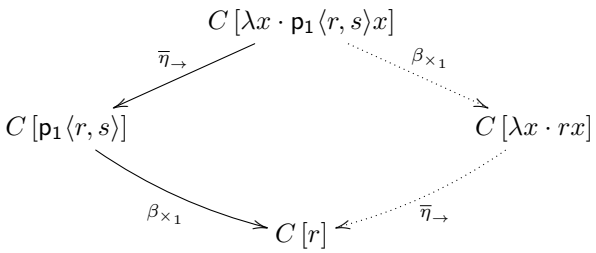


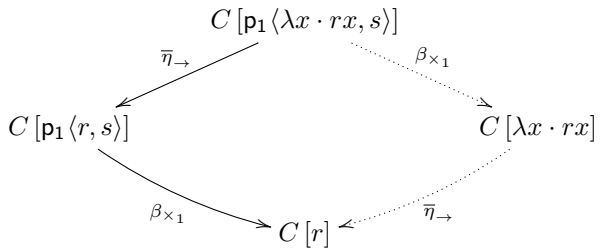


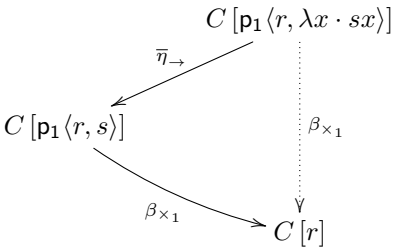




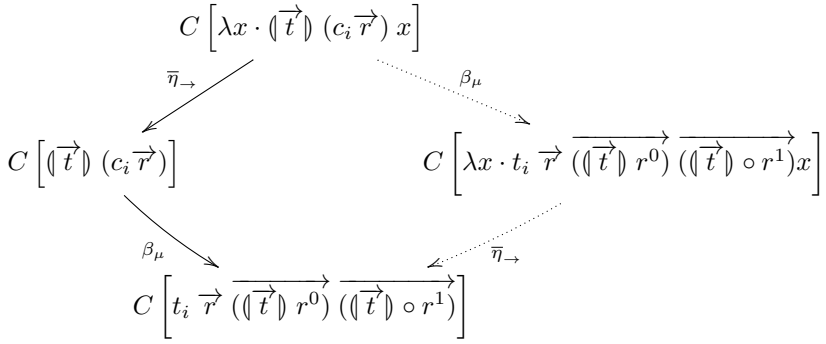
- $\beta_{\times_1}$ (the same for $\beta_{\times_2}$) :



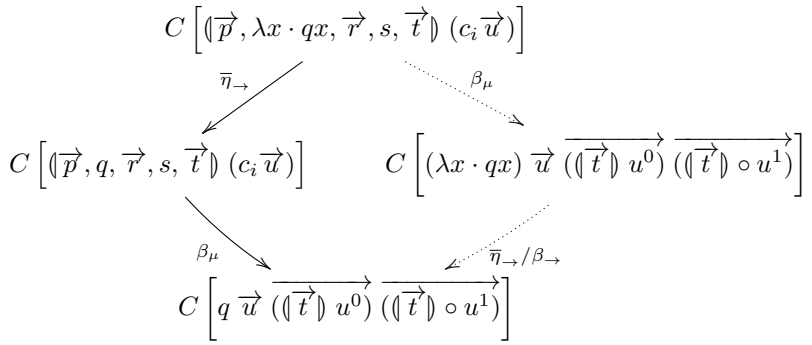




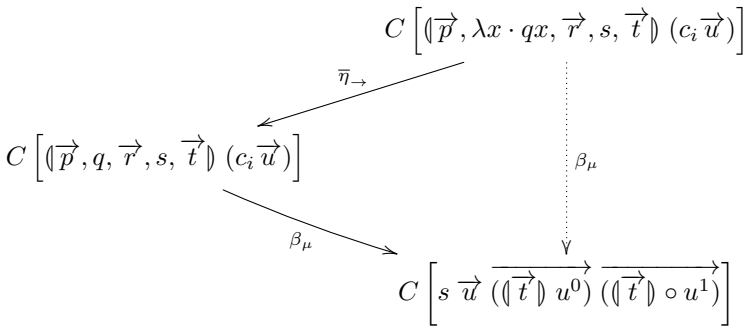
- β_μ : for this reduction, one case is more complex.
  - We may simply have:



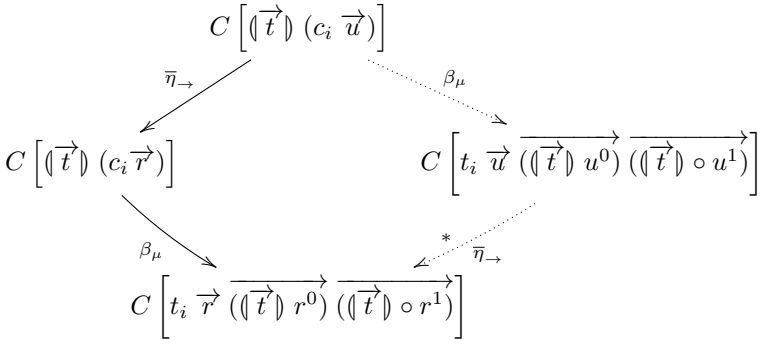
  - Or the rewriting step may happen on a recursion terms:
    - If q the ith recursion term, we end up with:



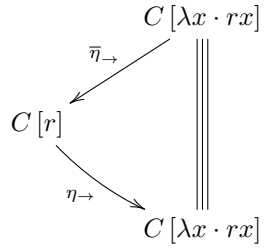
∗ Otherwise



– Finally, the rewriting step may concern one out of arguments, writing $\overrightarrow{r} = (\overrightarrow{p}, q, \overrightarrow{s})$ and $\overrightarrow{u} = (\overrightarrow{p}, \lambda x\cdot qx, \overrightarrow{s})$, which is likely to be parametric or 1-recursive:



- $\eta_{\times,1}\chi$: in $C\,[r]$, r cannot be contracted as it is not of functional type.
- $\eta_{\to}$:



□

APPENDIX B. PROOF OF LEMMA 4.5

Lemma 4.5. $\beta\eta\chi \vDash \Downarrow$.

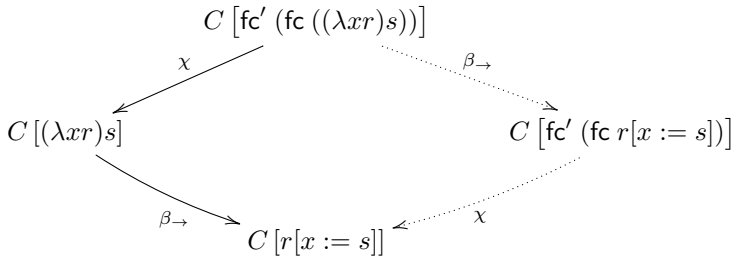
Proof. We define a condition P on terms s.t. 1-recursive arguments be in maximal abstracted form. One has:

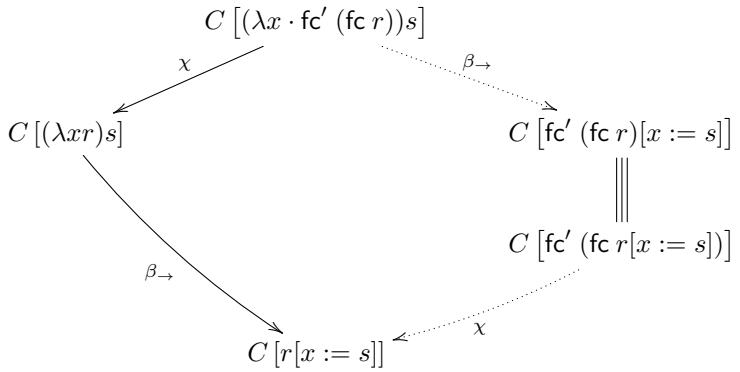
- $\eta_{\to} \subsetneq \beta\eta$;
- $\beta\eta \vDash \Downarrow$ by Theorem 4.1;
- $\chi \vDash \Downarrow$ by Lemma 4.3;
- $\eta_{\to}$ realises P as 1-recursive arguments are not in applicative position;
- $\eta_{\to}$ can be inserted in $\beta\eta\chi$ w.r.t. $\overline{\eta}_{\to}$ by Lemma 4.4 and $\eta_{\to} \vDash \Downarrow$. Hence, by Lemma 2.2, $\overline{\eta}_{\to}$ prosimulates and echoes $\beta\eta\chi$.

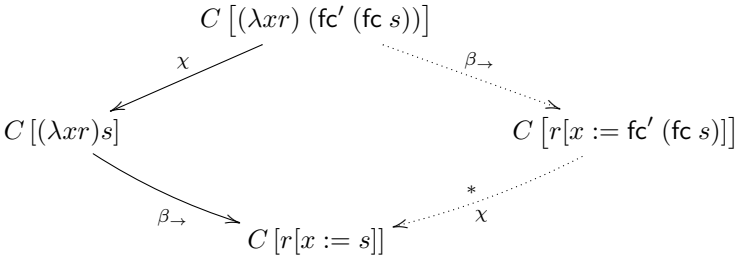
Moreover, as $\eta_{\to}^{-1} \subseteq \overline{\eta}_{\to}$, we know that for any pair $(t, t') \in \eta_{\to}$, we have indeed $(t', t) \in \overline{\eta}_{\to}$. Using the Pre-Adjusted Adjournment Lemma, we are left with proving that χ can be adjourned w.r.t. $\beta\eta$ under condition P

As χ-reduction is context insensitive and is compatible (Lemma 4.2), the contexts $C\,[]$ we consider, focussing on possible overlaps, have a simple form.

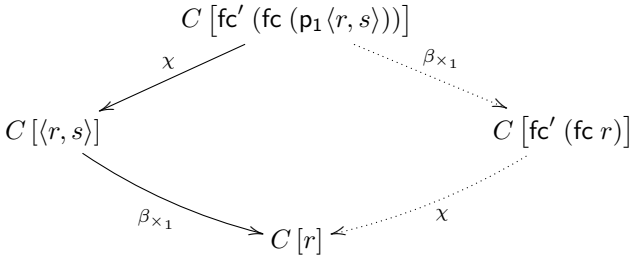
- $\beta_{\to}$:

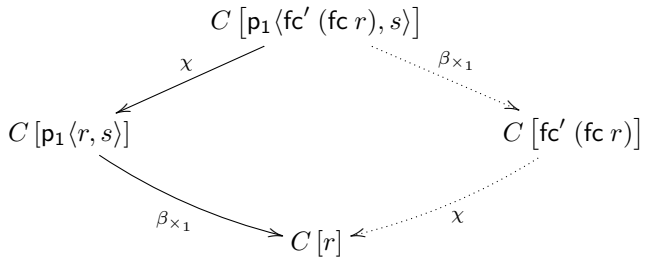


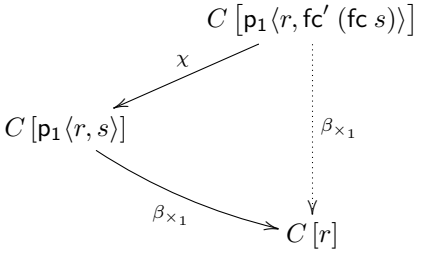




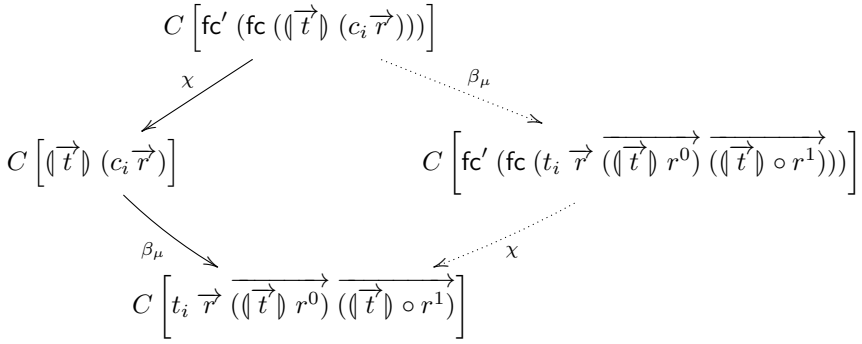
- $\beta_{\times_1}$ (the same for $\beta_{\times_2}$) :



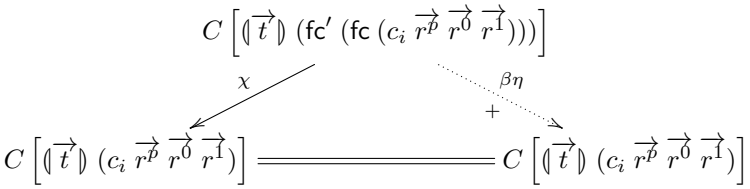




- β_μ : for this reduction, one case is a little bit more complex.
  - We may have:

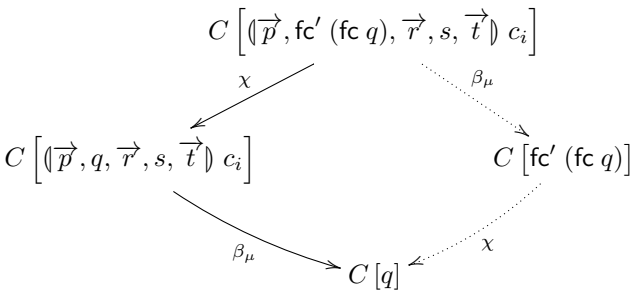


  - Recall that, because of the condition P, 1-recursive arguments are in maximal abstracted form, hence the following diagram:

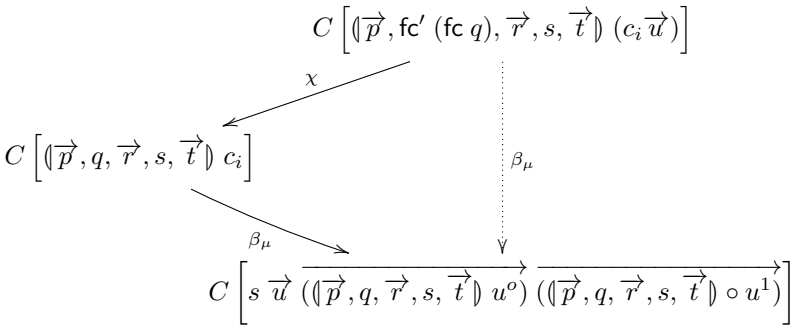


  - The χ-redex can be a recursion term:

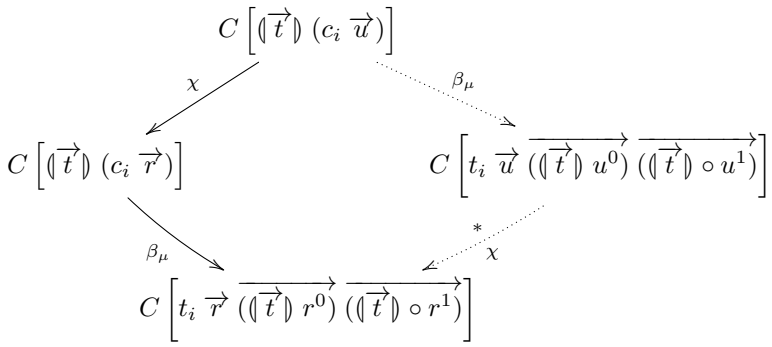
∗ If $\mathsf{fc}'\ (\mathsf{fc}\ q)$ is the ith term, then we necessarily have $\overrightarrow{u} = \varnothing$:



∗ Otherwise, if it is s, for instance:



– Finally, the χ-redex can be an argument, in which case we have the following diagram, writing $\overrightarrow{r} = (\overrightarrow{p}, q, \overrightarrow{s})$ and $\overrightarrow{u} = (\overrightarrow{p}, \mathsf{fc}'\ (\mathsf{fc}\ q), \overrightarrow{s})$:



- For η-reduction, no interesting overlap is possible as r is necessarily of inductive type in $\mathsf{fc}'\ (\mathsf{fc}\ r)$ and therefore cannot enjoy a η-reduction.

□

Appendix C. Proof of Lemma 4.12

Lemma 4.12. $\beta\eta\theta' \vDash \Downarrow$.

Proof. We define a condition P on terms s.t. these are in $\beta_{\overline{n}}$-normal form. We have:

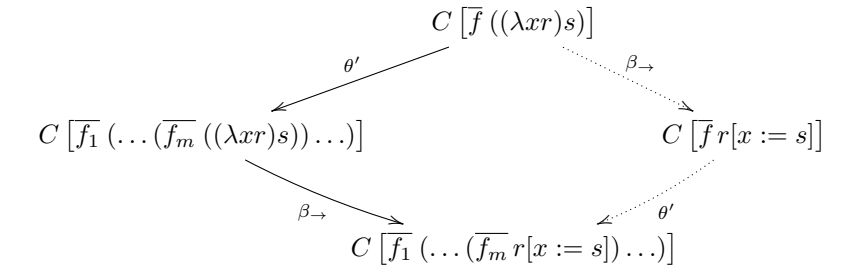
- $\beta_{\overline{n}}\theta' \subsetneq \beta\eta$;
- $\beta\eta \vDash \Downarrow$ by Theorem 4.1 ;
- $\theta' \vDash \Downarrow$ by Lemma 4.9 ;
- $\beta_{\overline{n}}\theta'$ obviously realises P (in fact, $\beta_{\overline{n}}$ is enough) ;

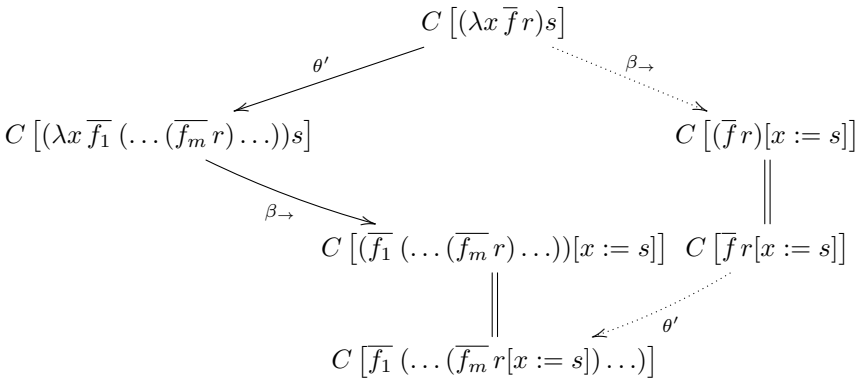
- $\mathcal{S}_f$ prosimulates and echoes $\beta\eta\theta'$ by Lemma 4.11.

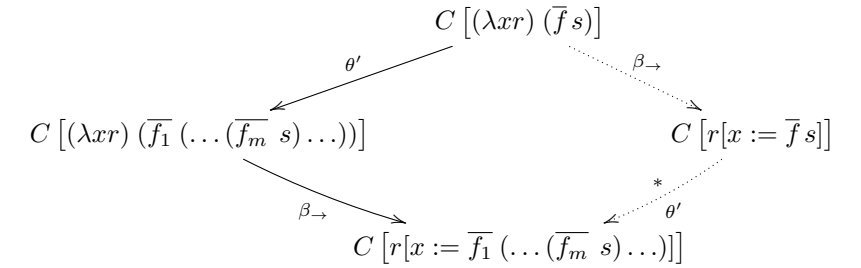
By the Pre-Adjusted Adjournment Lemma (Lemma 3.1), we just need to show that θ' is adjournable w.r.t. $\beta\eta$ under condition P.

As θ'-reduction is compatible and context insensitive (Lemma 4.8), the contexts $C\,[]$ that we consider, focussing on the possible overlaps, enjoy a simple form.

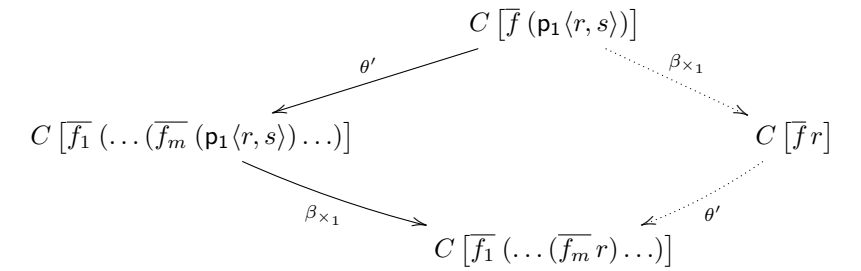
- $\beta_{\to}$:

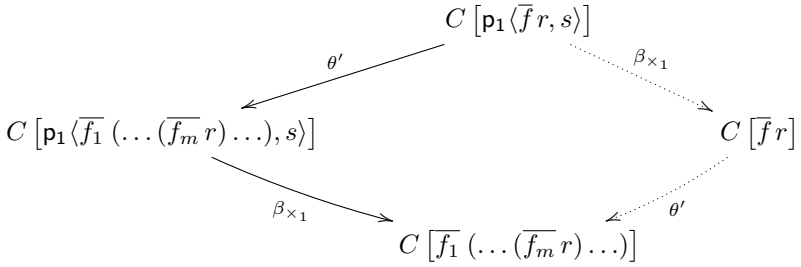


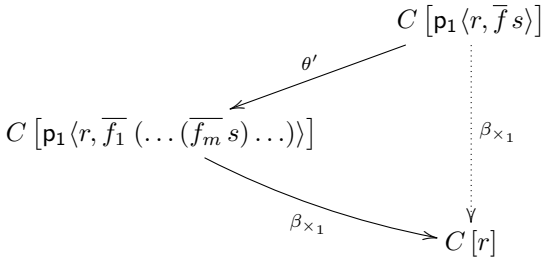




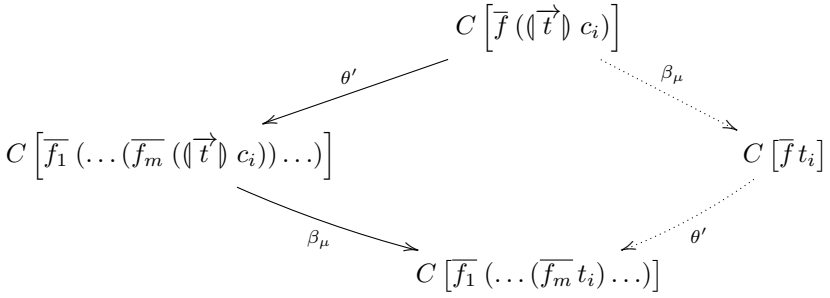
- $\beta_{\times_1}$ (the same for $\beta_{\times_2}$) :



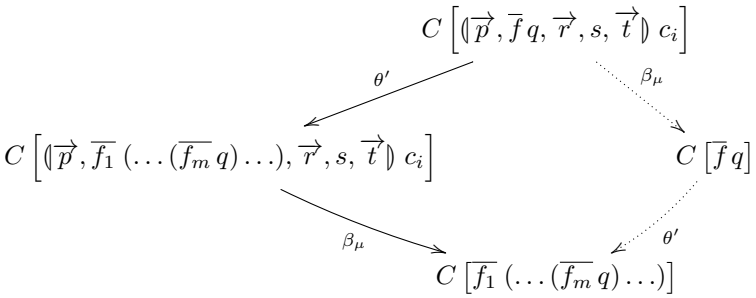




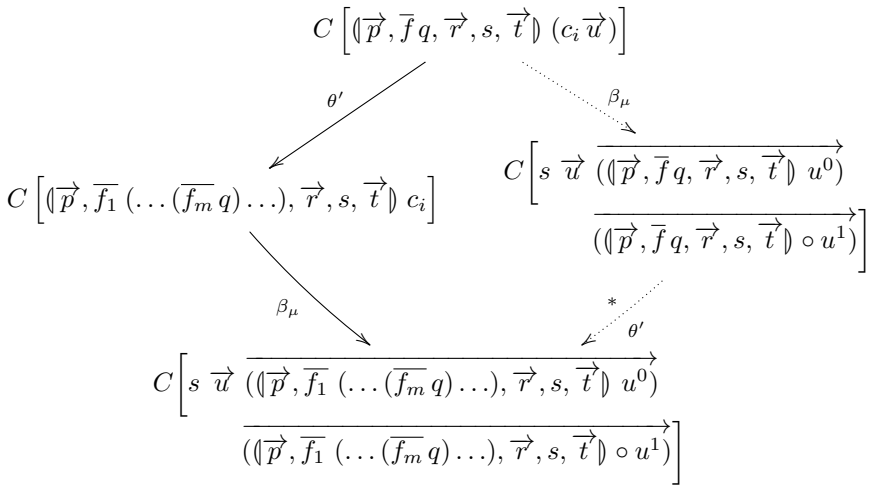
- β_μ. For this reduction, the situation is a bit more complex:
  - Notice first that it is not possible to have a context of the form $C\left[(\!|\overrightarrow{t}|\!)\ (\overline{f}\,c_i)\right]$ because it would not be in $\beta_{\overline{n}}$-normal form. However, we may have:



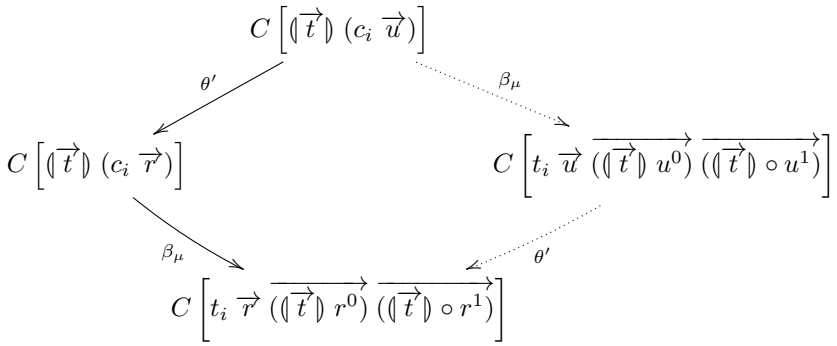
  - Otherwise, the θ'-redex can appear as a recursion term.
    - Either $\overline{f}\,q$ is the ith recursion term, in which case we have:



∗ Otherwise this term can be s :



– Finally the θ'-redex can appear as a *parameter* of a constructor. Write $\overrightarrow{u} = (\overrightarrow{p}, \overline{f}\,q, \overrightarrow{s})$ and $\overrightarrow{r} = (\overrightarrow{p}, \overline{f_1}\ (\ldots(\overline{f_m}\,q)\ldots), \overrightarrow{s})$:



• For η, there is no interesting overlap because of typing issues.

□

ACKNOWLEDGEMENT

The author wishes to thank Sergei Soloviev for useful advice and Freiric Barral for interesting discussions; as well as the anonymous referees for their detailed and fruitful comments.

REFERENCES

[Baader and Nipkow, 1998] Baader, F. and Nipkow, T. (1998). *Term Rewriting and All That*. Cambridge University Press, New York.

[Bachmair and Dershowitz, 1986] Bachmair, L. and Dershowitz, N. (1986). Commutation, transformation, and termination. In Siekmann, J. H., editor, *Proceedings of the Eighth International Conference on Automated Deduction (Oxford, England)*, volume 230 of *Lecture Notes in Computer Science*, pages 5–20, Berlin. Springer-Verlag.

[Barendregt, 1984] Barendregt, H. P. (1984). *The Lambda Calculus - Its Syntax and Semantics*. North-Holland, Amsterdam.

[Chemouil, 2005] Chemouil, D. (2005). Isomorphisms of simple inductive types through extensional rewriting. *Mathematical Structures in Computer Science*, 15(5):875–915.

[Chemouil and Soloviev, 2003] Chemouil, D. and Soloviev, S. (2003). Remarks on isomorphisms of simple inductive types. In Geuvers, H. and Kamareddine, F., editors, *Electronic Notes in Theoretical Computer Science*, volume 85. Elsevier.

[Di Cosmo, 1995] Di Cosmo, R. (1995). *Isomorphisms of Types: From λ-Calculus to Information Retrieval and Language Design*. Progress in Theoretical Computer Science. Birkhäuser, Boston, MA.

[Di Cosmo, 1996a] Di Cosmo, R. (1996a). A brief history of rewriting with extensionality. *In* Fairouz KAMAREDDINE, ed., *International Summer School on Type Theory and Rewriting*. Slides.

[Di Cosmo, 1996b] Di Cosmo, R. (1996b). On the power of simple diagrams. In Ganzinger, H., editor, *Proceedings of the 7th International Conference on Rewriting Techniques and Applications (RTA-96)*, volume 1103 of *LNCS*, pages 200–214, New Brunswick, NJ, USA. Springer-Verlag.

[Di Cosmo and Kesner, 1993] Di Cosmo, R. and Kesner, D. (1993). Simulating expansions without expansions. Technical Report RR-1911, INRIA.

[Di Cosmo and Kesner, 1996a] Di Cosmo, R. and Kesner, D. (1996a). Combining algebraic rewriting, extensional lambda calculi, and fixpoints. *Theoretical Computer Science*, 169(2):201–220.

[Di Cosmo and Kesner, 1996b] Di Cosmo, R. and Kesner, D. (1996b). Rewriting with extensional polymorphic lambda-calculus. *Lecture Notes in Computer Science*, 1092.

[Doornbos and von Karger, 1998] Doornbos, H. and von Karger, B. (1998). On the union of well-founded relations. *Logic Journal of the IGPL*, 6(2):195–201.

[Flegontov and Soloviev, 2002] Flegontov, A. and Soloviev, S. (2002). Type theory in differential equations. In *VIII International Workshop on Advanced Computing and Analysis Techniques in Physics Research (ACA'02)*, Moscou.

[Geser, 1990] Geser, A. (1990). *Relative Termination*. PhD thesis, Universität Passau, Passau, Germany.

[Girard et al., 1988] Girard, J.-Y., Lafont, Y., and Taylor, P. (1988). *Proofs and Types*. Cambridge Tracts in Theoretical Computer Science 7. Cambridge University Press.

[Lengrand, 2005] Lengrand, S. (2005). Induction principles as the foundation of the theory of normalisation: Concepts and techniques. Technical report, PPS laboratory, Université Paris 7. available at `http://hal.ccsd.cnrs.fr/ccsd-00004358`.

[Matthes, 2000] Matthes, R. (2000). Lambda calculus : A case for inductive definitions. Lecture notes for ESSLLI'2000 *(European Summer School in Logic, Language and Information)*.

[Soloviev and Chemouil, 2004] Soloviev, S. and Chemouil, D. (2004). Some algebraic structures in lambda-calculus with inductive types. In Berardi, S., Coppo, M., and Damiani, F., editors, *Proc.* TYPES*'03*, volume 3085 of *Lecture Notes in Computer Science*. Springer.

[Terese, 2003] Terese (2003). *Term Rewriting Systems*. Number 55 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press.